A Thesis on

OPTIMIZATION OF STEEL TRUSS USING MACHINE LEARNING

Submitted for partial fulfillment of award of

MASTER OF TECHNOLOGY
Degree in

STRUCTURE ENGINEERING

By
Mohd Shafat

(Roll No. : 1801431012)

Under the Supervision of
Mr. Tabish Izhar
Assistant Professor

Department of Civil Engineering
Integral University,
Lucknow–226026(U.P)
2020

DECLARATION

I declare that the research thesis entitled "**Optimization of Steel Truss using Machine Learning**" is the bonafide research work carried out by me, under the guidance of **Mr. Tabish Izhar,** Assistant Professor, Department of Civil Engineering, Integral University, Lucknow. Further, I declare that this has not previously formed the basis of an award of any degree, diploma, associate-ship or other similar degrees or diplomas, and has not been submitted anywhere else.

Date:

Place: Lucknow

Mohd Shafat

Roll Number

1801431012

Department of Civil Engineering

Integral University, Lucknow

CERTIFICATE

Certified that the thesis entitled "**Optimization of Steel Truss using Machine Learning**" is being submitted by **Mr. Mohd Shafat (Roll no. 1801431012)** in partial fulfillment of the requirement for the award of the degree of Master of Technology (Structures) of Integral University, Lucknow, is a record of candidate's work carried out by him under my supervision and guidance.

The results presented in this thesis have not been submitted to any other University or Institute for the award of any other degree or diploma.

Mr. Tabish Izhar

Assistant Professor

Department of Civil Engineering

Integral University, Lucknow

ACKNOWLEDGEMENT

I would like to thank and express my sincere appreciation to my thesis guide Mr Tabish Izhar for his limitless support and guidance. A year ago he introduced me to Artificial Intelligence and encouraged me to learn about its importance and its implementation in Civil Engineering. This thesis is nothing more than the implementation of what I learned while working with him. I would also like to thank Head of Department Prof. Dr Syed Aqeel Ahmad who has been very supportive to research students like me and has always strived to provide resources needed in quality research like MATLAB software package.

I would like to take this opportunity to thank Mr Mohd Sadat for his help and support. I am indebted to his suggestions and encouragement while writing this thesis. Finally, I would like to thank my family members, especially my parents and my wife for being a source of immense support and encouragement during this research work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATION

$f_n()$	function
m	member of truss
j	joint of truss
γ^e	unit weight of the material
L^e	length of each member
A^e	cross sectional area of truss
L	lower bound
U	upper bound
k	student number
i	iteration number
j	design variable number
T	teacher
M	mean
F	fitness function
Δ	minimum residual
c	connection or node number
X	design vector
σ	stress
δ	deflection
$\emptyset_\sigma$	stress penalty
$\emptyset_\delta$	deflection penalty
φ	total penalty
$N_{analysis}$	number of analysis
W_{avg}	average weight
α	intercept
β	gradient
TLBO	Teaching Learning based optimization
ML	Machine learning
LR	Linear Regression
(so)	structure optimization function

ABSTRACT

Modern structural optimization came in existence in the 1900s which began adopting gradient-based optimization. They had their fair share of drawbacks which led to their replacement by meta-heuristic algorithms. Meta-heuristic algorithms were capable methods which were inspired by nature and its processes such as particle swarm optimization, ant colony optimization, evolution-based algorithm, genetic algorithm, harmony search, and teaching learning-based optimization. To make these algorithms more effective modified versions were introduced in the last decade. The common drawbacks of these algorithms were constraint violation, lack of learning from experiences and requirement more iteration to reach a desired solution. In a nutshell, it can be said that they lacked smart computing.

Introduced in the 1950s Artificial Intelligence has been emerging in various fields. In the last decade, it has emerged in the field of Earthquake Engineering, Structural Engineering, and Structure Health Monitoring System. Aforementioned fields have used AI methods such as pattern recognition, deep learning, and machine learning for optimization. In this study machine learning was used to optimize the truss structure by reducing the search space and using adaptive analysis. The result of this study showed that the machine learning algorithm took less structural analysis while giving lightest design with no constraint violation. Result of this study showed that ML can be successfully applied in the structure design and optimization.

CHAPTER-1

INTRODUCTION

Optimization: The basic definition of structure in mechanics is an assemblage of different materials to sustain the load. Optimization is the process or method to find the best possible solution for a given situation or problem. The best possible solution can be maximization or minimization of a given function of the problem. So structure optimization can be defined as the process to make the assemblage of materials to sustain load in the best possible way. This best can differ with the aim of application and objective. In terms of a structure optimization, we want to find the structure which can transmit the load from the space region to fixed support in the best possible way. The best possible way could be to make the structure as light as possible by minimization of weight or maximization of structure strength by keeping the structure stiff, in both conditions the aim and approach are different. By making light structure we are making a cost-effective structure and by making stiff structure we are aiming for the structure that can withstand large values of loads under extreme circumstances where the intensity of load and forces on the structure change very rapidly as well as a very high value. Every optimization problems have a limiting factor that is called constraints; these constraints are the control variable that keeps the optimization process in a boundary of safe limits. Constraints are one of the main components of the optimization process, optimize structure is considered safe only if the constraint violation is minimum.

In the optimization of structure two methods are followed.

The iterative-intuitive method, in this method specific design is chosen and the requirement is checked with the optimal design. To attain the optimal design redesign is done until an optimal design is attained. In the border picture, it is a hit and trial method which keeps on changing at every iteration without following any specific logic or conceptual design method

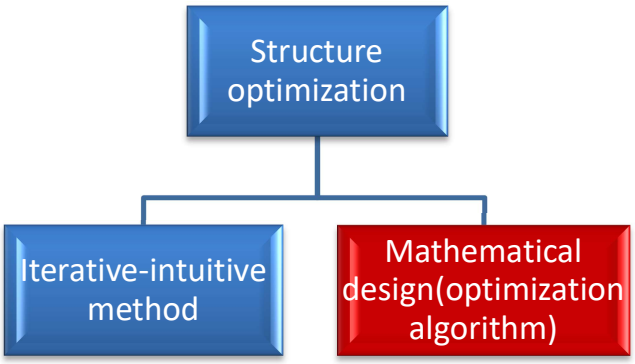


Figure 1. Structure optimization

because of that this method is generally very time taking process and is applied to simple problems. At a very large scale or complex problem using this method becomes very cumbersome and becomes impossible to use.

Mathematical design, in this method an optimization problem is formulated after the conceptual design is attained. The requirement of the design which acts as a constraint and under those constrain best possible ways is formulated to give the optimum result. To attain the optimum result an algorithm is formulated by studying and inspiring from a specific logic, these algorithms are called optimization algorithm, it is further classified as stochastic and deterministic methods which are discussed in detail in chapter 2. In this study, the mathematical design method of optimization is used to make the developed algorithm applicable to complex problems.

1.1 Mathematical form in an optimization problem

Some functions and variables are always present in any type of optimization.

Objective function (f): A function used to classify designs. For every possible design, function f returns a number which indicates the goodness of the design. Usually, f is chosen in such a way that a small value is better than a large one (a minimization problem and Vice a Versa in maximization problem). Generally, f is a measure of weight, displacement in a given direction, effective stress, or even cost of production.

Design variable (x): A function or vector that describes the design, and which can be changed during optimization. It may represent geometry or choice of material when it describes the geometry, it may relate to a sophisticated interpolation of shape or it may simply be the area of a bar or the thickness of a sheet.

State variable (y): For a given structure, i.e., for a given design x, y is a function or vector that represents the response of the structure. For a mechanical structure, response means displacement, stress, strain, or force. This variable is called

constraints that keep the structure safe and determinate in case topology optimization.

The general structural optimization problem takes the form:

$$(so)\begin{cases} minimize/maximize\ f(x,y) with\ respect\ to\ x\ and\ y \\ subject\ to \begin{cases} behavioural\ constraints\ on\ y \\ design\ constraints\ on\ x \\ equlibrium\ constant. \end{cases} \end{cases} \quad \text{Eq. (1.1)}$$

A generalized form of the above equation with multiple objectives is called multi-objective or criteria optimization:

$$minimize/maximize\ \{f_1(x,y), f_2(x,y), \dots\dots, f_n(x,y)\} \quad \text{Eq. (1.2)}$$

Where n is the number of objective functions, and all the constraints are the same as for(so). The above function is not a standard optimization function as the function will not be optimized for the same value of *x* & *y*. So as optimizers we try to reach Pareto optimality [1].

1.2 Types of optimization

The structural optimization is usually done in 3 ways of size optimization, shape optimization, and topology optimization.

Size optimization: In this method of structural optimization the size of the individual member of the structure is changed. The thicknesses of the members are generally reduced to minimize the cross-sectional area of the member while satisfying the design constraint as shown in fig 2(d). The minimization of the cross-sectional area also reduces the mass of the structure that is why size optimization is often termed as mass or weight optimization. The length and density of the members are kept fixed and thickness is changed at iteration, the number of members and the shape of the overall structure is not changed.

Shape optimization: In this method of optimization the size of an individual member is changed by changing the length and thickness of the member. Optimization in this form is attained by changing the shape and modifying the boundaries but keeping the number of node connections and number of elements fixed. In this approach, the best possible location of the node is determined to serve the purpose. The shape and size optimization is the most prominently used method as structural integrity and assemblage are not disturbed hence determinacy of the structure is not affected as shown in fig 2(c).

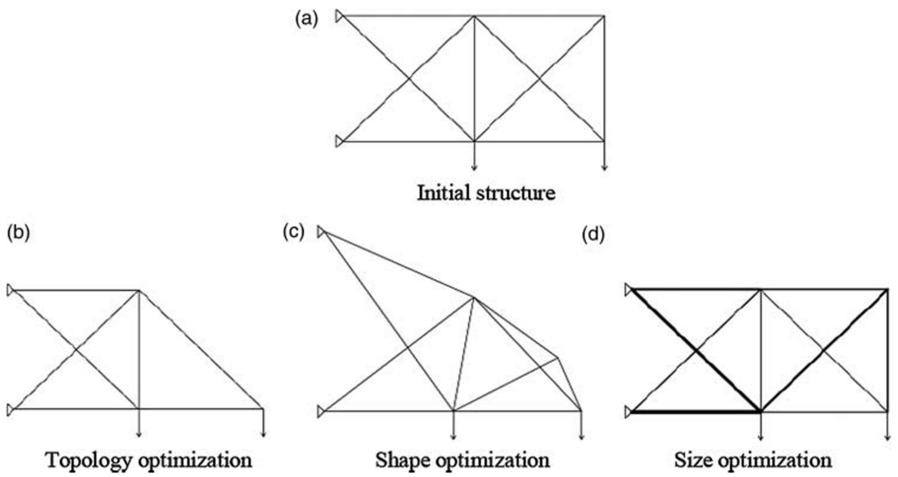


Figure 2. Different types of optimization: (a) Initial design, (b) Topology optimization, (c) Shape optimization, (d) Size optimization

Topology optimization: this is the most versatile form of optimization. In this method topology of the structure is changed by adding or removing certain members without losing the stability of the structure, for example in truss optimization area of certain members are taken as zero i.e. removing that member from the topology of the truss and changing the connectivity of nodes and thus changing the topology of members to get an optimal result as in fig 2(b). The design constraint in the topology optimization is considered more seriously as the violation of constraint is very common, usually, in the other two types of optimization two constraints stress and

displacement are enough to keep the structure safe but in this case, apart from two constraints, frequency constraints are also considered. This approach is very aggressive and requires high computational effort [1].

1.3 Truss

A truss is a structure that is an assemblage of beams and other elements of various shapes. These beams and members are called truss members; their combination forms a rigid structure. Truss members can be further be classified into different categories. Members under the tension, these members are on the bottom and are called as bottom chords. Members under the compression, these members are on the top and are called a top chord. The internal members are called webs and the areas under webs are called panels. Analysis of the truss is done by following two force member concept. The concept of this analysis is that truss can only take pure tension or compression without any bending or shear load. The joints of the nodes are pin-jointed, avoiding rigid joints. In other words, a truss can withstand the load purely by the axial resistance of its members. To maintain the axial resistance two conditions are to be met:

1. Joints between the members should be frictionless, can be pin-jointed.
2. All the loads must be applied at the joints and not on the members directly.

The assemblage of the members of the truss is not done in any arbitrary fashion rather it should be statically stable, if the structure is not statically stable due to requirement of assemblage then static and kinematic indeterminacy is determined for which analysis is done by either by joint method or section method.

Static indeterminancy

	External	Internal	
2D Truss	$r_e - 3$	$2j = m + 3$	Eq. (1.3a)
3D Truss	$r_e - 6$	$3j = m + 6$	Eq. (1.3b)

Kinematic Indeterminancy

2D Truss	$2j - r_e$	Eq. (1.3c)
3D Truss	$3j - r_e$	Eq. (1.3d)

Where *j* is the number of joints, *m* is the number of members in the truss and r_e is the external reactions ehich depend upon the type of support (fixed, hinged, roller).

The analysis method used in the work is based on method of joints which is sloved by flexibility matrix method.

Truss optimization is focused on finding a design that minimizes the structural weight while satisfying stress and displacement constraints. When the truss geometry is fixed the objective of the optimization is to select a cross-sectional area for each member so that the weight is minimized while stress and deflection constraints are met.

Weight optimization takes a general form of the following relation based on Eq.(1.1).

$$Minimum\ w = \sum \gamma^e L^e A^e \qquad \text{Eq. (1.4)}$$

Subject to:

$$\sigma^{lower} \leq \sigma^e \leq \sigma^{upper}$$
$$\delta^{lower} \leq \delta^e \leq \delta^{upper}$$
$$A^{lower} \leq A^e \leq A^{upper}$$

Where,

w is the weight of the truss which is composed of *N* members and for each member *e*

γ^e is the unit weight of the material for each member

L^e is the length of each member

A^e is the cross-sectional area of truss for each member

The truss design must satisfy limits on member stress σ^e and deflection δ^e at each connection. Limit on the cross-sectional area, stress and deflection are given by *lower (L)* and *upper(U)* boundaries [26].

Stress constraints and deflection constraints are addressed as a penalty function. Formulation of the penalty function is done by addressing the σ^e and δ^e in each member of the truss is compared with the maximum allowable stress $\sigma^{Lower,Upper}$ and maximum allowable deflection $\delta^{Lower,Upper}$ of each connection *c* of the truss. The penalty function is used to account for the infeasible truss design. The objective function is multiplied with the cumulative penalty function that is proportional to the amount of stress and deflection constraints violation. This method generates a slightly heavy objective function (structural weight) but always keep the violation in check, heavy structural weight is the drawback which has been eliminated using ML algorithm.

The stress penalty function $\emptyset_\sigma^e$ for each member, *e* is defined as

$$if\ \sigma^{Lower} \leq \sigma_e \leq \sigma^{Upper}\ , then\ \emptyset_\sigma^e = 0 \qquad \text{Eq. (1.5)}$$

$$if\ \sigma_e < \sigma^{Lower}\ or\ \sigma_e > \sigma^{Upper}, then\ \emptyset_\sigma^e = \left|\frac{\sigma_e - \sigma^{Lower,Upper}}{\sigma^{Lower,Upper}}\right| \qquad \text{Eq. (1.6)}$$

The total stress penalty $\emptyset_\sigma^e$ for a truss design, *k* is:

$$\emptyset_\sigma^k = \sum_{e=1}^{N_m} \emptyset_\sigma^e \qquad \text{Eq. (1.7)}$$

The penalty function for deflection in the *x, y, z* directions $\emptyset_{\delta x}^c, \emptyset_{\delta y}^c, \emptyset_{\delta z}^c$ are computed at each connections *c* is defined as

$$if\ \delta^{Lower} \leq \delta_{c(x,y,z)} \leq \delta^{Upper}\ , then\ \emptyset_{\delta(x,y,z)}^c = 0 \qquad \text{Eq. (1.8)}$$

$$if\ \delta_{c(x,y,z)} < \delta^{Lower}\ or\ \delta_{c(x,y,z)} > \delta^{Upper}, then\ \emptyset_{\delta(x,y,z)}^c = \left|\frac{\delta_{c(x,y,z)} - \delta^{Lower,Upper}}{\delta^{Lower,Upper}}\right| \qquad \text{Eq. (1.9)}$$

The total deflection penalty $\emptyset_{\delta}^{c}$ for a truss design, k is:

$$\emptyset_{\delta}^{k} = \sum_{c=1}^{N_c} \left[\emptyset_{\delta x}^{c} + \emptyset_{\delta(y)}^{c} + \emptyset_{\delta(z)}^{c} \right] \quad \text{Eq. (1.10)}$$

The total penalty φ^{k} for truss design k is then equal to the sum of stress and deflection with a positive penalty exponent ε, defined as:

$$\varphi^{k} = (1 + \emptyset_{\sigma}^{k} + \emptyset_{\delta}^{k})^{\varepsilon} \quad \text{Eq. (1.11)}$$

The fitness function F^{k} is the product of the weight of truss design k and its total penalty:

$$F^{k} = \varphi^{k} W^{k} \quad 1.12$$

1.4 Optimization algorithm

The mathematical approach to optimization is done by the formulation of an algorithm for conceptual design and an algorithm is known as an optimization algorithm. The optimization algorithm is classified into two categories where first is the deterministic method under which we have a gradient-based optimization algorithm and the second is termed as the stochastic method under which we have heuristic and meta-heuristic algorithm.

Gradient-based optimization algorithm: the optimization method is based on gradient ascent or gradient descent to reach to the nearest best solution. These methods are generally iterative method that relies on the information of gradient. Some examples of gradient-based methods are Sequential Linear programming, Sequential Quadratic programming. The shortcoming of this approach has been discussed in the literature overview.

Heuristic and meta-heuristic algorithms are the most adopted method in recent literature because of their simplicity of code and implementation. These methods are generally nature-inspired and are developed by mathematical representations of

natural processes. Examples of the meta-heuristic algorithm are Particle Swarm Optimization (PSO), Genetic Objectives (GP), Teaching-Learning Based Optimization (TLBO), Symbiotic Organism Search (SOS), Jaya algorithm, Water Wave Optimization (WWO), Big Bang Big Crunch (BB-BO), Whale Optimization Algorithm (WOA), Evolution Algorithm (EA) and their respective enhanced version are used in recent works of literature, these meta-heuristic algorithms are population-based search methods. In this study teaching-learning based optimization has been used with machine learning. The machine-learning algorithm was used to reduce the search space and then TLBO was applied if needed.

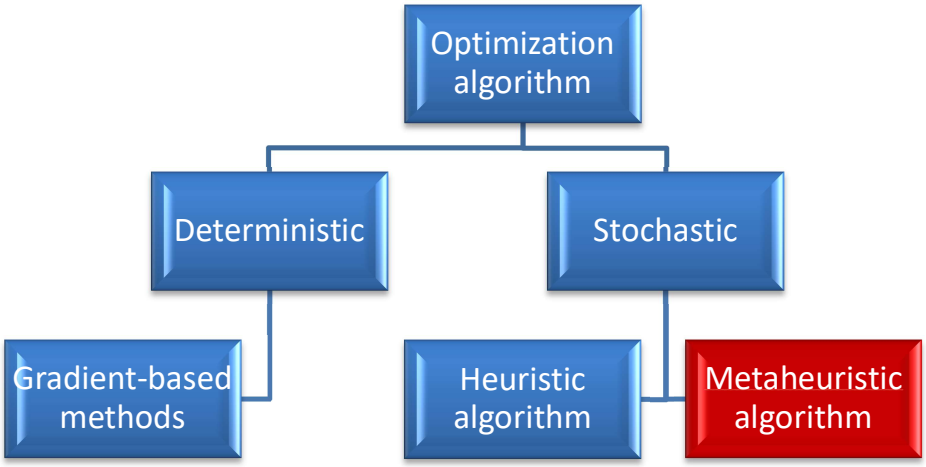


Figure 3. Classification of the optimization algorithm

1.4.1 Teaching learning based optimization

The TLBO is a population-based meta-heuristic search method like HS, ACO, PSO, BB–BC, and ABC. The TBLO method presents a mathematical model for optimization problems based on the simple teaching process. The primary objective of the TLBO method is to improve the average performance of individuals in a population. It is assumed that the distribution of grades in a given class of students

follows a normal distribution, in such a class, a more qualified teacher will provide better student outcomes as measured by a higher mean value. At iteration i, the best student is selected as the teacher Ti and then the teacher shares his or her knowledge with other students to improve overall performance or mean of the class from Mi to Mi+1. The process is repeated until the best teacher is obtained. An analogy between the TLBO and the optimization of truss structures is established in the following way: a class is considered as a population which contains truss designs, a learner in a class denotes a truss design in the population, a design variable represents a subject taught to the student, the grade of a student denotes the weight of truss design, the teacher is the truss design with the lowest weight in the population. In addition to the teacher-student interaction, there is collaborative learning among students. In the TLBO algorithm, both of these classroom interactions are implemented in two different consecutive processes: a Teacher Phase that simulates the influence of a teacher on students; and a Learner Phase that models the cooperative learning between students

Teaching phase: The mathematical equation of the teaching phase is written as

$$X_{new}^{k} = X_{old}^{k} \pm \Delta \qquad \text{Eq. (1.13)}$$

$$\Delta = T_F \times r|M - T| \qquad \text{Eq. (1.14)}$$

Where,

X^{k}_{new} denotes the design variable for the k^{th} design vector, T_F is the teaching factor with a random number r within range of [0,1], M is the mean of class, T is the state of the teacher. Δ denotes the difference between the teacher and class mean (update) for each design variable and it should be selected in such a way that the student always moves towards the teacher. The teaching factor T_F in Eq. (1.14) is the only adjustable parameter in the modified TLBO algorithm and is used to increase the size of the local search space around each student. Rao et al. [46] presented data to indicate that a value of T_F = 2 is appropriate to balance both the exploration and exploitation aspects of the search in the Teacher Phase same value has been used in this study. Both teacher and mean values play important roles in directing the

population through the search space. Since TLBO is an iterative process, at the end of each teaching cycle, the role of the teacher is updated to the current best student in the class. As define in Eq. (1.15), the computation of the mean is important to establishing the scale of the search. The equation of mean which has been used here was given by Rao et al.[46] in the original algorithm as.

$$M = \frac{1}{N}\sum_{k=1}^{N} X^k \, , N \; is \; the \; size \; of \; population \qquad \text{Eq. (1.15)}$$

Learning phase: the procedure of the learning phase can explain as two random student p and q are selected form the class in such a way that $p \neq q$and their fitness is evaluated to see if $F^p < F^q$ then,

$$\begin{gathered} X_{new}^p = X_{old}^p + r\left|X_{old}^p - X^q\right| \\ \text{otherwise} \\ X_{new}^p = X_{old}^p + r\left|X^q - X_{old}^p\right| \end{gathered} \qquad \text{Eq. (1.16)}$$

Here r is a random number within a range $\{0,1\}$. The position of student p is adjusted toward q within the search space for $F^p > F^q$ else the position of p is adjusted away from q. in either case p tries to improve itself until the constraints are not violated.

1.5 Artificial intelligence

Artificial Intelligence (AI) is a branch of computer science concerned with making computers act more like human beings. It was first introduced by in workshop held in Dartmouth college. It is a computational method attempting to simulate human cognition capability through symbol manipulation and symbolically structured knowledge bases to solve engineering problems that defy solution using conventional methods. It is a specialized system to understand intelligent entities, construct them, and make the process of decision making simple, quick, and efficient.

In general, there are two types of machine intelligence: hard computing and soft computing methods. Hard computing, which is based on binary logic, crisp systems,

and numerical analysis, requires a precisely stated analytical model and is capable of producing precise answers. Soft computing differs from conventional computing in that, unlike hard computing, it can deal with ambiguous and noisy data, incorporates stochastic information, and allows parallel computations. Soft computing is based on fuzzy logic, neural nets, and probabilistic reasoning; where the methods can evolve their programs and yield approximate answers [2]. Soft computing is commonly considered a synonym of computational intelligence (CI). CI or soft computing can be expressed by the capability of a computer to learn a specific task from sample data or experimental observation. Mathematical or conventional modeling is useless in many complex real-life problems due to factors such as the complexity of the processes for mathematical reasoning, uncertainties during the process, and the stochastic nature of the process. The set of nature-inspired computational techniques defining CI provides solutions for such problems CI uses a combination of supplementary techniques such as artificial neural networks, fuzzy logic, learning theory, evolutionary computing, and probabilistic methods, and is capable of solving and approximating nonlinear problems while introducing human knowledge into the areas of computing. There are other machine intelligence apart from CI and SI such as Big data and Data mining. All these are the subset of AI to serve the different purposes of different fields. Related to civil engineering, we need to study about deep learning, ANN, fuzzy logic, and pattern recognition [47].

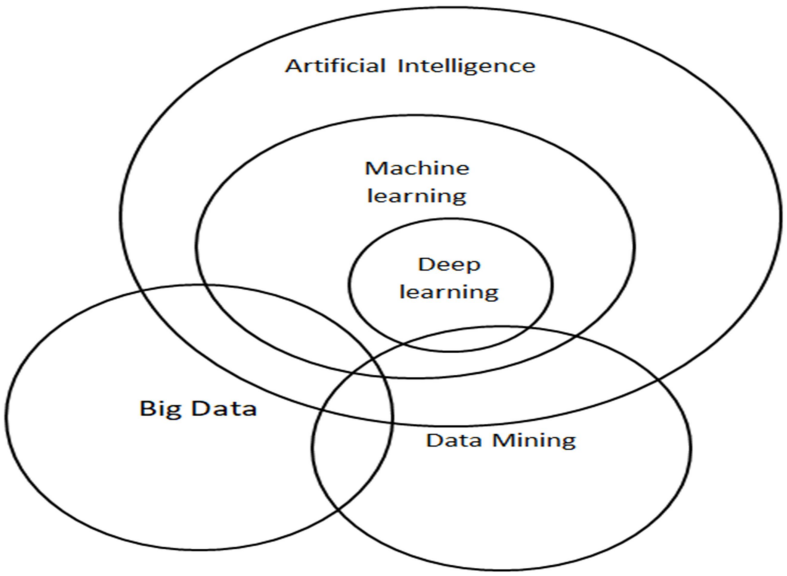


Figure 4. The interrelation of different intelligent computational technique

In the field of civil engineering, it covers a vast area for human benefits especially in engineering design and optimization, structure health monitoring system, construction management and can solve complex problems to the level of experts by imitating the experts. The traditional methods for design, modeling, optimizing complex structure systems and manual observation of activities are difficult, time-consuming, and prone to error, so, AI helps in automated data collection and data analysis techniques to improve several aspects of construction engineering and management for productivity assessment, safety management, idle time reduction, prediction, risk analysis, decision-making and optimizing construction costs. AI methods that are emerging in recent areas are namely Machine learning (ML), Pattern recognition (PR), Deep learning (DL), Swarm Intelligence, Artificial neural network (ANN), fuzzy logic, decision tree [2]. Some of them are discussed below.

1.5.1 Pattern recognition

Pattern recognition (PR) is a technique in which the main goal is to classify objects into several classes or categories. The objects, depending on the applications, could be images, signals, handwriting, speech, or measurements to be classified [35, 36]. In

PR, a pattern is represented by a set of features. Concepts from statistical decision theory are used to establish decision boundaries between pattern classes. The recognition system in PR consists of two modes, namely learning (training) and classification (testing). There has been a growing interest in the application of pattern recognition (PR) to structural engineering for purposes such as structural health monitoring (SHM)/damage detection, earthquake engineering and seismic design, structural reliability, structural identification, and performance evaluation. The most common use of PR in structural engineering has been for SHM and damage identification.

1.5.2 Deep learning

Deep learning (DL), a branch of machine learning, is composed of networks that can learn unsupervised from unstructured/unlabeled data. DL architecture aims to learn the feature representation of the input data. DL is based on deep neural networks, i.e., neural networks with more than one hidden layer. In such an architecture, increasing the number of layers results in a deeper network. Examples of DL architectures include convolutional neural networks (CNNs), recurrent neural networks (RNNs), auto encoders, deep belief nets, etc. there has been a growing interest in the use of deep learning, e.g., convolutional neural networks (CNNs) for structural engineering applications, mainly in structural health monitoring (SHM). The application of CNNs is very new in the field of SHM and damage detection. CNN's within the context of SHM is defined as learning and extracting optimal features and classification using learned features. CNN's are primarily designed for two-dimensional signals (e.g., images, video frames, etc.), thus leading to an efficient image recognition method. Therefore, CNNs are categorized and used as vision-based SHM techniques in which dataset are images captured at various stages of the structure being monitored.

1.5.3 Machine learning

Machine learning (ML) is a major subfield of artificial intelligence dealing with the study, design, and development of algorithms that can learn from the data itself and make predictions using learned data. It refers to the capability of computers to learn without being explicitly programmed. ML-based models can be predictive or

descriptive to achieve knowledge from the data. The scope and potential of ML are much more general than other AI methods, although it is a subset of AI and used in various disciplines of engineering and technology. The application of machine learning methods has been increasingly adopted over the last decade for modeling real-world problems concerning structural engineering. This is because of their enormous capacity to capture relations among input and output data that are nonlinear or complicated to formulate mathematically. The first uses of ML techniques in structural engineering have dealt with problems such as the development of management tools for structural safety and information acquisition for the design of steel members. In general, ML methods have been used for SHM and damage identification, optimization, performance evaluation, structural reliability, and reliability assessment, and structural parameter identification e.g., modeling material properties of concrete. Among these, SHM and concrete property modeling are the uses to attain the most attention whereas there has been very little work in optimization during the last decade. The application and advantages of machine learning in optimization has been left to explore [2] [37].

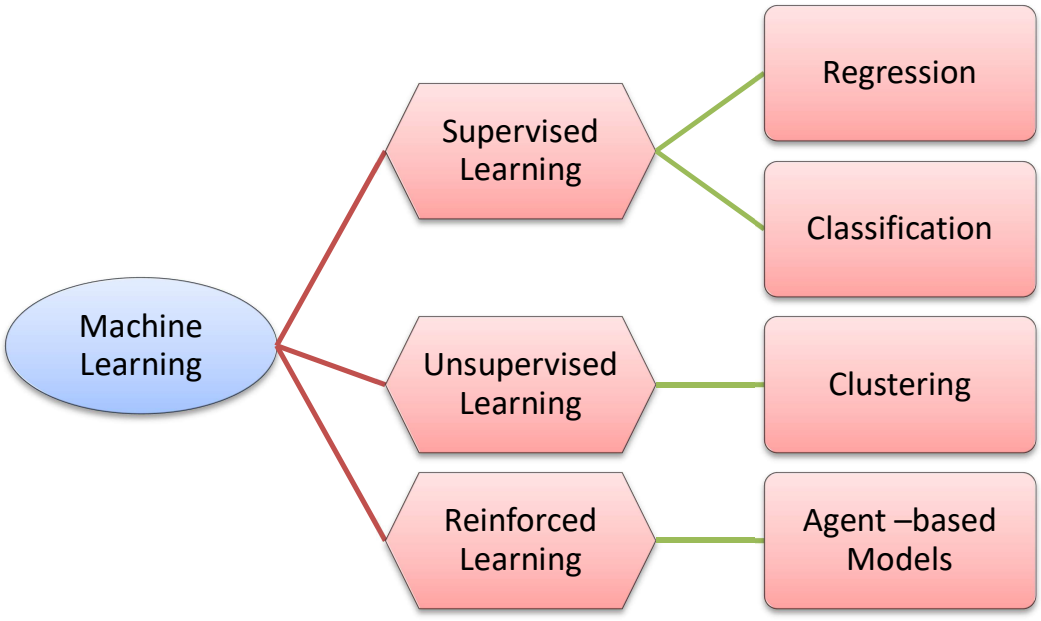


Figure 5. Machine learning methods

Machine learning methods are categorized into 3 forms as illustrated in figure 5.

Reinforcement learning: In reinforcement learning, or learning with a critic, no information is given regarding the desired category signal or explicit goals. Reinforcement algorithms are forced to learn optimal goals through trial and error. To maximize the model's performance, reinforcement learning allows an algorithm to determine the ideal behavior within a specific context. The machine algorithm receives a numerical reward as a reinforcement signal encoding the success of an action's outcome. The goal for the algorithm is then to learn to select actions maximizing the accumulated reward over time [2]. The classification of regression methods is shown in figure 6.

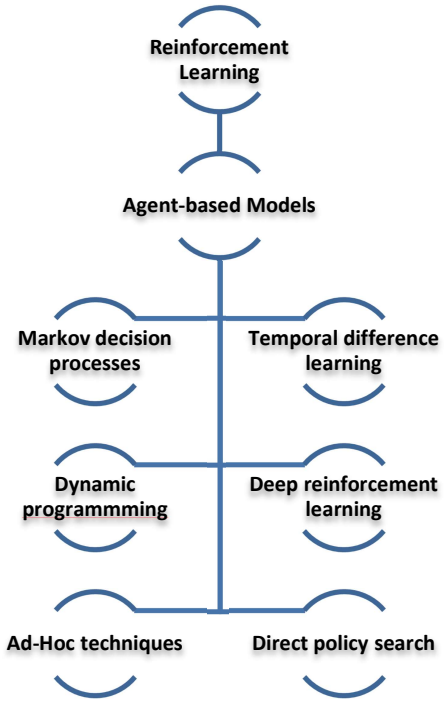


Figure 6. Reinforcement learning methods

Unsupervised learning: Unsupervised learning is the training of the machine algorithms using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of the machine is to group unsorted information according to similarities, patterns, and differences

without any prior training of data. Unlike supervised learning, no teacher is provided that means no training will be given to the machine. Therefore the machine is restricted to find the hidden structure in unlabeled data by it-self, methods of unsupervised learning have been shown in figure 7.

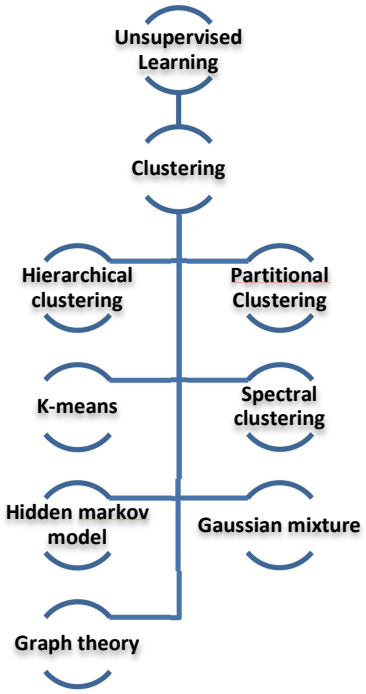


Figure 7. Unsupervised learning methods

Supervised learning: Supervised learning as the name indicates the presence of a supervisor as a teacher. Supervised learning is a learning in which we teach or train the machine using data that is well labeled which means some data is already tagged with the correct answer. After that, the machine is provided with a new set of examples (data) so that a supervised learning algorithm will analyze the training data (set of training examples) and produces a correct outcome from labeled data. Supervised learning is done by 2 methods, regression, and classification. Under regression, we have linear regression, nonlinear regression, neural network, etc. For classification, we have the Bayesian method, k-nearest neighbor, decision trees metric learning, etc are shown in figure 8.

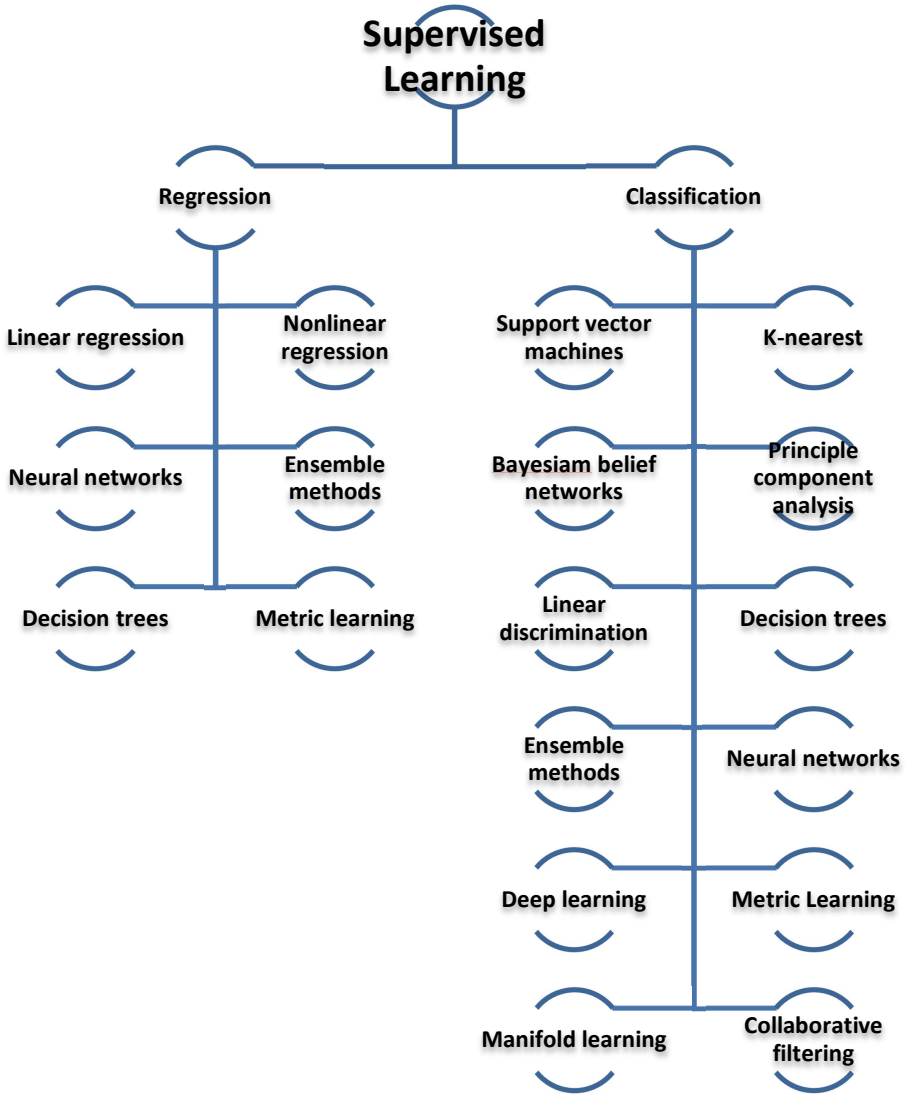


Figure 8. Supervised Learning methods

In this study, to improve the search space of the teaching-learning algorithm machine learning in the form of linear regression by fitting the curve of extrapolation of the predicted value. The basic equation for linear regression fitting is as follows.

$$y = \alpha \pm \beta x \pm \Delta \qquad \text{Eq. (1.17)}$$

The relationship between α and β is unobserved and the model that is used to underlie that is called the linear regression model. The goal is to find the best-suited value of α_{new}^{i} and β_{new}^{i} from the predicted value with minimum residual Δ which is subjected to minimum for $\alpha_{new}^{p}, \beta_{\text{new}}^{p}, \alpha_{new}^{q}, \beta_{new}^{q}, \alpha_{new}^{r}, \beta_{new}^{r}$ from initial.

$$\Delta = min \sum_{i=1}^{n} (y_{new}^{i} - \alpha - \beta x_{new}^{i})^2 \quad \text{Eq. (1.18)}$$

$$\beta_{new}^{i} = \frac{\sum_{i=1}^{n}(x - x_{new}^{i})(y - y_{new}^{i})}{\sum_{i=1}^{n}(x - x_{new}^{i})^2} \quad \text{Eq. (1.19)}$$

There are some predicted values which have do not fit and some fit perfectly, the number of perfectly fitted values tells the applicability and effectiveness of the linear regression model. The predictive model generates the initial value on the same principle as of the TLBO between lower and upper limits while following the constraint violation.

1.6 Objective

The objective of this thesis can be classified into three parts.

- The first objective of this study is to develop a MATLAB script data for the geometry and properties of selected truss and use that data in the MATLAB model which consist of a script of ML and meta-heuristic algorithm.
- The second objective of this study is to successfully optimize the selected model using the developed model.
- The third objective is to validate the result of the developed ML model with previously available literature to validate the application of ML in the field of optimization that could evolve to be applied in a more complex function.

CHAPTER-2

LITERATURE REVIEW

Structure optimization problems are generally multi-objective in nature which can be solved by many different algorithm approaches such as metaheuristic algorithm, derivative-free optimization algorithm (heuristic algorithm) and gradient-based methods. Metaheuristic algorithms are the most adopted method in recent literature because of their simplicity of code and implementation. These algorithms are inspired by natural phenomena. Examples of metaheuristic algorithm are particle swarm optimization (PSO), genetic objectives (GP), teaching-learning based optimization (TLBO), symbiotic organism search (SOS), Jaya algorithm, water wave optimization (WWO), big bang big crunch (BB-BO), whale optimization algorithm (WOA), evolution algorithm (EA) and their respective enhanced version are used in recent works of literature [34].

Particle swarm optimization is the algorithm that is inspired by natural phenomena of the social behavior of animals such as fish schooling, insect swarming, and bird flocking. Whale optimization, ant colony optimization, artificial bee colony optimization, and firefly optimization algorithm are swarm-based optimization algorithm. This algorithm is used for the global optima for various types of arbitrary problems. It was introduced by Kennedy and Everhart in 1995. It is a simulation of graceful and unprecedented choreographs of birds flock, the simulated resulted came to be a stochastic population-based evolutionary computer algorithm [7]. In the study by **Zhengtong et al.** [4], which aims to optimize the layout of a truss (topology optimization) using the improved version of particle swarm optimization. They also used a concept of using multi-material in a truss structure by using the special type of steel in different parts of the truss. Steel with different strength and mass were used in different parts which were under different loading condition. To determine the efficiency of the developed algorithm zhengtong et al. performed a trial on 15, 39, and 47 bar elements problems. In 15-bar element problem zhengtong et al. compared their result with a previously developed version of PSO and variants of evolutionary algorithms and found that their algorithm gave 3-10% better result and in some cases comparable result in terms of optimization. In the case of the 47-bar element problem, a comparison was done with the result of a previously developed version of PSO and variants of evolutionary algorithms and it was found that the present

algorithm gave 2.7 to 7% better result and in some cases comparable result. In the case of the 39-bar element, their algorithm performed better than evolutionary algorithms. In an overall comparison, the result obtained was found to be the better-optimized structure in terms of mass and hence cost, having optimum strength. In the study by **Mortazavi et al.** [5] a newly developed optimization algorithm namely an interactive search algorithm (ISA) was used and the result was compared by previously existing meta-heuristic algorithms. The interactive search algorithm (ISA) was developed by closely studying and use of an affirmative feature of PSO with the pairwise knowledge sharing mechanism of TBLO. The best of both algorithms were merged. The hybrid algorithm was tested on previous problems and validated by other pre-developed meta-heuristic algorithms. The result of this research was compared with BB-BC, ABC, TLBO, mTLBO, colliding bodies optimization and its variant (CBO, eCBO), flower pollination optimization (FPA), and previously developed version of PSO for 25, 72, 200 bar truss problem and 582 bar tower problem. It was found from the comparison that ISA gave 6.5-8% better results than other algorithms as well as their parent algorithm. It was also noted that ISA was better than its parent algorithm in terms of computation cost and searchability. ISA was able to avoid getting stuck on local minima as effectively as one of its parent algorithm TLBO and optimize the elements of the truss as effectively as iPSO. The best of both algorithms were used to get a better hybrid algorithm. **Kaveh et al.** [6] too used a hybrid algorithm with a combination of particle swarm optimization and chaos theory. It was the multi-phase algorithm in which the first phase was to control the parameter value by chaos theory and the second phase was for local search ability by particle swarm analysis. For spatial 24-bar truss problem elements were divided into 8 groups, the result of CSP was compared with previously developed versions of PSO, ACO, BB-BC, GA, and CAA. CSP took 350 iteration and 17500 analyses which were about 28% more than other algorithms, in this CSP outperformed PSO and BB-BC. For the spatial 72-bar truss, problem elements were divided into 16 groups, the result of CSP was compared with GA, PSO, SAHS, BB-BC, and outperformed GA. For planar 200-bar truss problem elements were divided into 29 groups, the result of CSP was compared with GA, self-adaptive harmony search, CMLPSA. CSP took 317 iteration and 31,700 analyses which were also 12% more

than others. For spatial 942-bar tower truss problem elements were divided into 59 groups and results were compared with GA, PSO, BB-BC, HBB-BC, and gave .072% to 29% better result than others. Optimization of a mass of truss structure was successfully done by **Gomes et al.** [7]. Frequency constraints which turn to be conflicting with mass optimization at lower bound were addressed by the use of particle swarm optimization. On comparing with other approaches with the solved examples it was found that for 10-bar, 37-bar, 72-bar truss problems present algorithm was not able to give as effective result as in the compared literature but when compared for 52-bar truss problem it was able to produce better result than other literary works. While swarm optimization was found to give heavier structure than compared literature, the results were acceptable from the engineering point of view and swarm optimization was found to advantageous to be working with population of random parameters and required less function evaluation as compared to other meta-heuristic algorithms used in previous literature works.

Ant colony optimization is a nature-inspired algorithm that is based on the process of an ant navigating towards the food source from their nest leaving the trail behind for other ants to follow. The intensity of trial defines the best result as the most intense trailed path will be most followed by other ant leaving individual trails. This phenomenon is used in the field of optimization by creating an artificial trial that can be followed by the latter population. The most trailed path is considered to lead towards the best result. In the research work by **Camp et al** in 2004 and 2005 [10] [11] same methods were used for designing space frame and steel frame. In comparison with the GA, TSA and other methods on various examples it was found that ACO gave better results.

Artificial bee colony optimization is the algorithm that is inspired by the behavior of female honey bee whose task is to forge the nectar from the food source. The female bee looks for the potential food source which represents the possible solution. Bees are divided into three categories, employed bee, unemployed bee and scout bee. When a bee finds a food source then she is termed as employed bee, which exploits the food source and shares the information of food source to unemployed bee. Unemployed bee either follows the employed bee on the basis of waggle dance or

search for food source. The scout bee abandons and then looks for new food sources. The food source with the most employed bee is termed as best solution. This process is iterative in nature and if no further improvement is seen than the food source is discarded. This process moves forward towards best solution in random fashion and converges at best solution. In the research work by **Sonmez** [12] artificial bee colony with adaptive penalty function was used on benchmark problem of 10-bar, 18-bar, 22-bar, 72-bar and 200-bar truss. A comparison of the result of this benchmark problem was done with harmony search, particle swarm optimization, annealing and ant colony optimization. It was found from the result that artificial bee colony gave structurally sound result and had global approach and does not require evaluation of gradient and constraint functions. This made artificial bee colony one of the preferred methods of choice.

Whale optimization is also a nature-inspired algorithm that is based on the swarm-based optimization in which the hunting behavior of humped back whale is simulated on the basis of two methods, shrinking circle and spatial bubble-net feeding maneuver. In the research work by **Kaveh et al** [13] a modified version of WO was used. A modification was done to standard version in way that if at the selected position the reliability was less than 50% then the new position was selected. An enhanced version of WO was used on 72-bar and 582-bar spatial truss and the results were compared with the result of particle swarm optimization and big bang-big crunch method and EWO was found to be the best robust method to give the lightest volume in an independent run.

The evolution of species is the natural process and evolutionary algorithms are inspired by the process of evolution and mutation which is based on the theory suggested by Charles Darwin. In this algorithm initial population is mixed with each other to produce a mutated population and the survival of the fittest that is the population that gives the best result is selected. The fitness of the organism also ensures the adaptation of the organism under different conditions. In the research work by **Bureerat et al.** [14] differential evolution algorithm was used on 10-bar, 25-bar, 72-bar, 200-bar truss benchmark problem. The result of this study was compared with artificial bee colony, TLBO, BB-BC and other meta-heuristic

algorithms. It was found that from the comparison that TLBO and the present algorithm performed better than other algorithms and gave structurally acceptable structure. In the research work by **Ahrari et al.** [15] evolution strategy was used on 18-bar and 77-bar truss benchmark problem and it was found that ES algorithm was performed with the local search and resizing of elements.

Genetic programming is the optimization algorithm that is based on nature's principle of genetic diversity. When an evolution algorithm is allowed to evolve on a large parameter then that strategy is termed as genetic programming. Programming is done by simulating the genetic tree as the binary program in which leaves and branches represent the functions and terminals. The binary operation, standard program, logic and custom functions are set under functions and constraints, variables and specific quantities are set under terminals. GP employs the crossover and mutation to explore the search space and find new solutions in every generation. Genetic programming works on Darwin's theory of survival of the fittest [10]. A weight and geometry optimization of truss-z system was done by **Zawidzki et al.**[16] by using NSGA. A comparison was done of weight and geometry optimization. Optimization of sizing and topology of truss was done by the minimization of mass in the study by **Assimi et al.** [17]. Minimization of mass results in the optimum cross-sectional area and connectivity of joints. The structure was kept kinetically stable under maximum allowable stress and deflection. The genetic programming was capable of identifying redundant members and joints. When compared to the results available in literature work on conventional method, GA, harmony search algorithm, swarm optimization, ACO and other meta-heuristic algorithms, the genetic programming gave relatively lighter truss members. The comparison was done for 10 elements with 6-joint truss problem in two cases with a difference of applied load and load joint method, it was found that for case 1 SOGP gave 0.01%-7.35% better result than other algorithms and for case 2 it gave 4.63%-9.49% better result. The comparison was also done for 17- elements problem and 39-elements problem respectively to get 0.21%-1.16% better result than compared method for 17-element problem and 0.10%-2.39% for 39-element problem respectively, however for 39-element SOGP gave 0.07% heavier than ACO. Improved version of genetic

programming was used in the literature work of **Togan et al.** [18]. The adaptive approach of genetic programming was used with improvement in penalty functions. The improved version of GP was tested on previously solved problems in the other works of literature to give a better result than the parent algorithm. It was able to get a global solution for small problems. A 112-bar dome was analyzed and compared with other variants of genetic algorithm, it was found that present study was able to give better results in terms of volume and area when 3 member groups were analyzed. A 200-bar element was analyzed by using different penalty functions that were obtained from a previous genetic algorithm to give similar results by a varying number of generations. The combined approach of two methods was also done by **Cazacu et al.** [19] in which optimization was done by GP and stress-displacement constraints analysis was done by the finite element method. FEM and GP were implemented in MATLAB. Results were compared with MATLAB toolbox GPLAB and other literary works. This approach gave much faster computational speed and better solutions. In the literature work by **Jenkins** [20] it was suggested that the implementation of GP was easy when compared to other methods of optimization.

Teaching-learning based optimization is the meta-heuristic algorithm which is a population-based search space method. It is the simulation of teaching-learning of class between the teacher and learner. It was first introduced by Rao[47]. The performance of the learner is dependent on the teacher and a good teacher produces a good class [14]. A modified TLBO algorithm approach was used by **Baghlani et al**. [21]. The efficient handling of constraint was used by mapping the whole population into feasible space. This modified approach enhanced the ability of developed algorithms to give feasible optimal design. There were two main objectives of this research first, to efficiently handle the constraint for which algorithm was modified and secondly, to reduce the number of analyses required for reaching the optimal solution. Along with this modified algorithm (TLBO-MS) two other modified algorithms which were modified by using two other constraint handling method namely, penalty function (TLBO-PF) and fly back (TLBO-FB) were studied. The result of TLBO-MS along with results of TBLO-PF and TLBO-FB were compared with different available works of literature with different approaches using various

examples. Comparison was done- for 10-bar planar truss under two cases, for 25-bar spatial truss, for 72-bar spatial truss, for 120-bar spatial truss with BB-BC, PSO and its derivatives, GA,ACO,ABC,CBO, improved version of harmony search, TBLO and its improved versions, to determine whether the optimal point given by TLBO-MS was structurally acceptable or not. It was found by the comparison that TLBO-MS gave structurally acceptable results which were either slightly heavier or lighter by other methods and also it did not give any constraint violation. The TLBO-MS performed better than TLBO-PF, TLBO-FB in terms of weight of members. When comparison was made on number of structural analyses required, TLBO-MS performed better than other compared algorithms and other variations of its parent algorithm (TLBO-PF, TLBO-FB). **Tejani et al.** [22] and **Savsani et al**. [23] both tried to implement the improved version of TLBO, HTS, WWO, PVS in their research and compare them of the basis of benchmark problems which were solved in the previous works of literature. Savsani et al. used the benchmark problem form previous literature to optimize the topology of 24-bar truss problem, 20-bar truss problem and 72-bar 3-D truss problem. While Tejani et al. performed benchmark problem to optimize topology of 10-bar, 15-bar, 25-bar and size and topology of 39-bar problem form literature works of other researchers on PSO, TLBO, BB-BC, HS, ABC, ACO, PVS and its improved version and other meta-heuristic algorithms. Both Savsani et al. and Tejani et al. got similar results in which improved version of PVS got better results followed by PVS and improved version of TLBO. In overall analysis it was found that improved versions of each algorithm performed better than their parent algorithm and gave structurally acceptable results when compared with other literature work. **Farshchin et al**. [24] used modified TLBO optimization algorithm. The modification was made by introducing the multi-class teaching and learning ability in two-stage process. In first stage multiple classes with learners were introduced and teaching was done and when a predefined level of learning is achieved then the best student from each class is taken and new class with these best performers is introduce and at second stage this newly formed class is used for local search. This two-stage modification increases the exploration capability which in turn increases the search efficiency of the algorithm. Modified algorithm was tested on benchmark problems that were present in the literature on PSO, GA, HS, FA,

CBA, improved version of BB-BC, swarm-based meta-heuristic algorithms and results were compared with the same. The benchmark problems were of 10-bar truss, 37-bar truss, 52-bar truss, 72-bar truss, 200-bar truss problems. As a result, it was found that for MC-TLBO performed 0.01% for 10-bar problem and 0.1% for 52-bar truss better than HS and 0.06% for 72-bar truss better than GS. For 37-bar truss and 200-bar truss topology optimization was done in which it performed comparable to TLBO and gave structurally acceptable results. In overall comparison the strength was MC-TLBO was to get at the optimal result in a fewer analysis than other compared literature methods and MC-TLBO proved to be faster than other meta-heuristic methods. **Camp et al.** [26] also used a modified version of TLBO; modification comes in way that search population was divided into the small group consisting of 75 members. This modification enhances the ability of algorithm to deal with population of large space and also increases the efficiency of algorithm. Benchmark problems of 10-bar cantilever truss, 25-bar space truss and 72-bar space truss. These benchmark problems were taken from the literature work of GA, ACO, HPSO, BB-BC, ABC and its improved version, PSO, HS and its improved version. On comparison of result it was found that for 10-bar cantilever truss and 25-bar space truss developed algorithm gave structurally acceptable result which was slightly heavier or lighter with compared literature. For 72-bar space truss modified TLBO produced lightest result than other compared methods in the compared literature. In overall result it was found that modified TLBO required less computational effort than other methods and was more efficient in terms of analysis required to reach the optimal design. Application of a teaching-learning based algorithm (TLBO) was done in **Degertekin et al.** [27]. It was applied to the examples in the literature and results were compared with them. It was found that the optimized truss produced by the TLBO was slightly heavy but required less structural analysis. TLBO was applied with the strength and displacement constraints provided in accordance with the AISC in the literature work by **Togan** [29]. The result found to be comparative to the other algorithms with independency on the number of parameters and has a simple numerical structure.

The symbiotic organism's search (SOS) algorithm is also a nature-inspired algorithm. The basic concept of this algorithm is the interdependency of the symbiotic organism. The simulation of this used in computer programs to find global optima. In the comparative study by **Tejani et al.** multi-objective modified adaptive symbiotic organism search (MOMASOS) was found to give best mean values when compared multi-objective adaptive symbiotic organism search (MOASOS), multi-objective symbiotic organism search (MOSOS), multi-objective ant colony system (MOASC), multi-objective adaptive search (MOAS) when compared in the analysis of a 10 bar truss, a 25 bar space truss, a 60 bar-ring truss, a 72 bar truss, a 942 bar tower truss respectively [30]. This study of Tejani was the extension of his earlier work in which comparison SOS and MSOS were done with other meta-heuristic algorithms and MSOS was found to be superior to others and its parent algorithm [31].

In the study by **Afshari et al.** [32] a comparison of 6 different algorithms was done, namely multi-objective gradient-based algorithm, multi-objective derivative-free optimization algorithm, multi-objective genetic algorithm, Non-dominated sorting genetic algorithm-III, multi-objective particle swarm optimization, and a random method for the optimal design of RC beam. The efficiency of a multi-objective gradient-based algorithm was found to be moreover heuristic and meta-heuristic algorithm.

JAYA algorithm is the algorithm which was developed recently by Rao et al.[46] it is the meta-heuristic approach towards the optimization. It is also a population-based search algorithm like TLBO. This algorithm is based on the concept that one should move towards the best result and ignore the worst result. The search process is to move towards the best design and move away from the worst design. This algorithm has no algorithm-specific parameters rather it has population and iteration as a parameter of optimization. Jaya algorithm moves towards the best design in every iteration. For the optimization of the 3-D structure, a study by **Tayfun et.al** [33] was done on a benchmark problem of dome-shaped 120-bar element truss using the Jaya algorithm with the frequency constraints. For the optimization of the 3-D structure, it is important that the algorithm should be developed under the constraint of frequency. The result of this study was compared and validated by other algorithms

such as CBO, PSRO, DPSO, SOS-ABF. The benchmark problem was also optimized in the previous works of literature. In the comparison of result it was noted that Jaya algorithm was found to be 01%-2% better than other algorithms and outperformed other algorithms in terms of efficiency. **Degertekin et.al** [34] used a modified version of the Jaya algorithm namely DAJA. This modified form of Jaya algorithm was under two constraint stress and displacement to perform a discrete optimization of the truss under discrete sizing, layout, and topology. In this study of Degertekin et.al the native behavior of the Jaya algorithm was changed to follow a descent path adjacent to every solution and provide good quality test results. Benchmark problems of 10-bar planar truss, 25-bar tower truss, 47-bar power line tower, 52-bar planar truss, 72-bar spatial truss, 200-bar planar truss, 942-bar spatial truss were compared with HPSO, DHPSACO, CSS, TLBO, CBO, AFA, WCA, IMBA, HHS, aeDE, SA, SGA, ECS, BI, GA, ESASS. The finding of the study showed that DAJA produced very comparable results without violating the optimization constraints. It was also noted from the result that modified JAYA took considerably less structural analysis run to give converging point or optimum result and proved itself to be more efficient than the best one available one i.e. TLBO. In a previous study by **Degertekin et.al** [35] Jaya algorithm was used on the widely available problems in the literature. The result this algorithm for benchmark problem of 200-bar planar truss, 942-bar spatial tower truss, 1938- bar tower for sizing-layout optimization of truss and 25-bar transmission tower, 45-bar planar truss and 47-bar power line truss for sizing optimization was pitched against the result of previous researchers methods of optimization which were HBB-BC-LS, BB-BC, HHS-LC, HHS, ABC-AP, SAHS, TLBO, MSPSO, CA, FFA, FPA, WEA, ES, GA, HPSSO, HPSACO to get a comparative analysis. This comparison was done over 3 different parameters that were minimum weight, standard deviation, and optimized weight. Jaya algorithm gave a very comparable result with other meta-heuristic algorithm and in some cases it was able to reduce the weight of truss beyond what other algorithms had achieved. The main strength of jaya algorithm was to reach the optimal point with less structural analysis. It was a point to note that in all the studies on the Jaya algorithm, the algorithm was developed in the MATLAB coding platform.

In the optimization of truss by means of topology optimization which gives a free form approach to designing an efficient structure layout, the optimization algorithm is known to give optimum results that are supported by various works of literature. It is also well known that oversimplification of underlying structure optimization formulation leads to the impractical and instable structural solutions. The optimization approach is commonly to minimize the liner stain energy or minimize the mass of the truss element. The optimized result comes out to be a thin member that cannot resist yielding or local buckling or a member with un-braced hinges that destabilized the structure. In the research work by **Nwe et.al** [36] it was pointed out that the optimized truss was unstable. In their work new algorithm was developed that incorporated the yield strength and stability limit constraints in the formulation of the optimization algorithm. The new algorithm was based on the previous work of Cheng & Guo that is the Epsilon Continuation Approach and Disaggregated Formulation by Achtziger. The newly developed algorithm was tested on a simple truss system and was validated by the above-mentioned work. The resulting algorithm gave the optimized truss that was stable under stress, local buckling, and global stability constraints, leading to solutions that were far more structurally relevant.

Machine learning is the ability of a computer algorithm to learn from previous experiences without being programmed further. In structure engineering repetitive analysis and designing are done for similar structures with defined acting forces and constraints. This pattern of iteration is followed in the optimization method. To overcome this, machine learning can be used as a tool for learning from previous results. In the study by **Zeynep et al.** [37] optimization of the space, the frame was done using a machine learning method. In this study, optimization was done with the use of the artificial neural network (ANN) and finite element analysis (FEA). To train the algorithm, an example structure was developed by the iterative method and these structures were optimized. The objective of the study i.e. to validate the optimization by machine learning method was achieved and this method itself proved to generate less error. **Lorenzo** [38] showed an approach to optimize the connection of steel structure using machine learning. It was found in the study of Lorenzo that

optimized connections were comparable with those of manual design while using fewer materials. Cost optimization of the reinforced concrete frame was done by **Bekas et al.** [39]. In this study, the optimization was performed by an evolutionary algorithm and the optimized result was subjected to a predictive modeling algorithm to develop a predictive model. The result of this study showed that the cost prediction of the optimal design of the structure is possible. In the research work by **Tamura et al.** [40] machine learning algorithm was used for the optimization of placement of bracings of steel frame building. In this research simulated annealing was used with machine learning to reduce the computational time and cost of optimization process. Machine learning algorithms used were binary decision tree (BDT) and support vector machine (SVM). Results of SA with and without machine learning algorithms were compared

2.1 Inference of literature review

1. The original form of any algorithm was mutated and improved to get a better result against other algorithms and their parent algorithm.
2. The trend to the development of all new algorithms has seen a significant dropping in recent literature. The improved version of the pre-existing algorithm was predominant over the development of the new algorithm.
3. The PSO was found to be lacking in control of initial parameter values as they were difficult to define. It was found that PSO showed premature convergence for the iterative process and was also found to be stuck at local optima for complex and high dimensional space.
4. In the original approach of genetic programming and other forms of evolutionary algorithm, it was found that they were very much dependent on the initial design. The optimal solutions generated by the genetic programming varied for every iteration.
5. When compared with other optimization methods TLBO required less iteration to reach the optimal result but it gave a slightly heavier result. It performed better for high dimensional space.
6. The gradient-based optimization method was found to susceptible to get stuck at local minima and was inefficient for high computational complex problems.
7. A combination of machine learning and an evolutionary algorithm can be used to keep a check on the violation of local and global constraints resulting in a safer structure while reducing the number of structural analyses [32] [36].
8. The hybridization of meta-heuristic and machine learning algorithms will be able to solve complicated problems with less iteration. With the use of hybrid algorithms learning of the meta-heuristic can be improved and it can create a new opportunity for global learning to algorithms [36-38].

CHAPTER-3

METHODOLOGY

3.1 **Selection of truss:** In the world of optimization, any new or modified optimization method is first applied, tested, and verified on truss and then on higher-order structure type. Every algorithm is initially developed for steel truss as it is one of the most research fields in structural engineering. Members of a truss are manufactured in a factory under a certain specific standard that is prescribed by the standard code of practice that makes their property uniform and enables the approach of optimization quite simple. For the study and development of the MATLAB model, we have selected a 10-bar cantilever truss, 25 bar space (3D) truss also known as a transmission tower, and a 72-bar space truss consisting of multiple stories.

3.2 **Analysis of truss:** Develop MATLAB script for the geometry and properties of the truss which will be imported as the data for the developed ML model

3.3 **Formulation of a machine learning model:** Machine learning model will be formulated in 2 different stages

3.3.1 Formulation of a Meta-heuristic algorithm for optimization of analyzed truss which will be based on the evolutionary algorithm (TLBO).

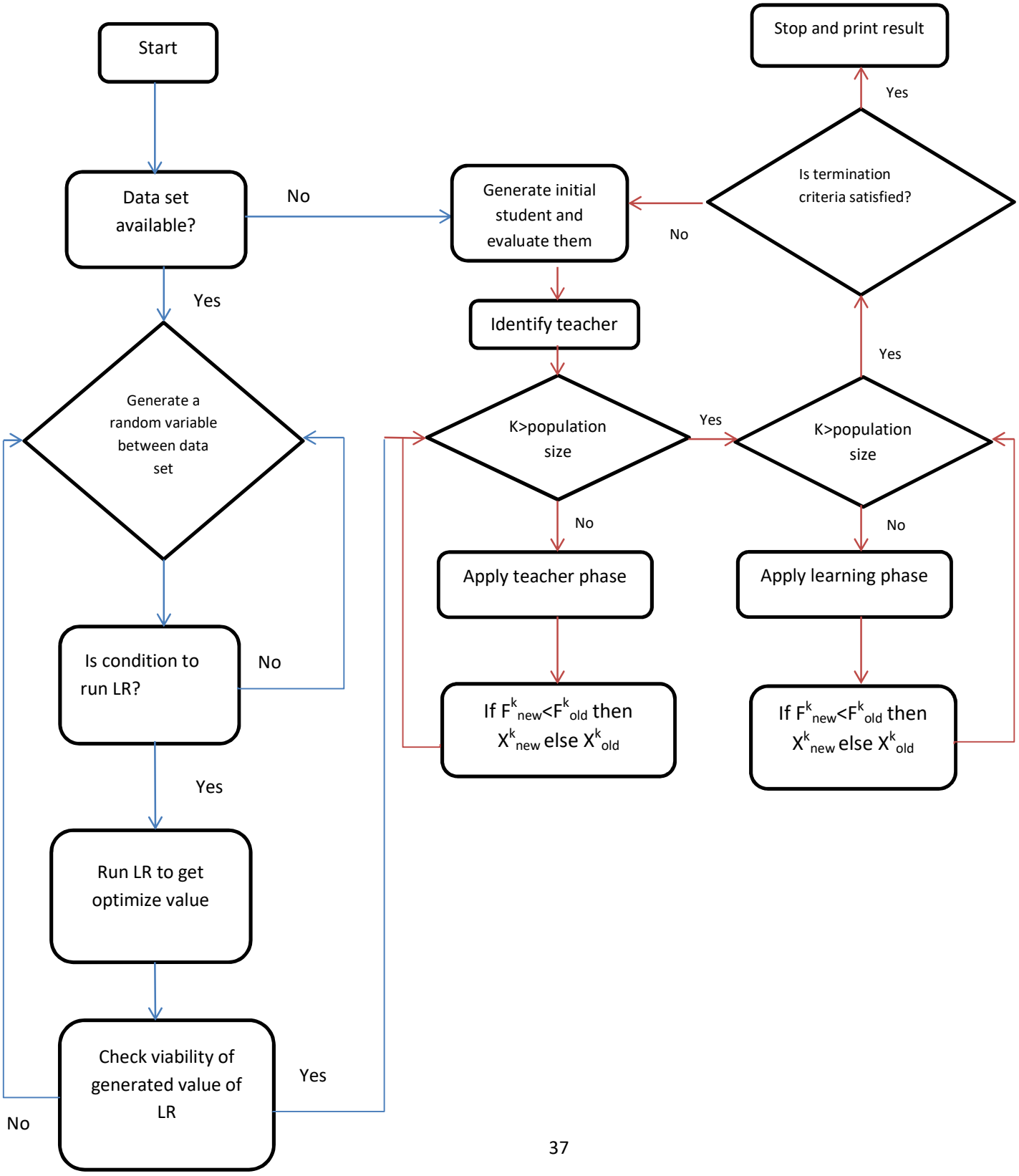
3.3.2 Formulation of a Machine learning algorithm for optimization of a truss. The machine learning algorithm will be used with a meta-heuristic algorithm in the optimization of selected benchmark trusses.

3.4 **Training of algorithm:** training of the ML model will be done for every truss problem whose training data are not available. The training process incorporated in the ML model which works along with the meta-heuristic algorithm

3.5 **Optimizing the truss and validating the result:** Optimizing and validating the result generated by the ML model with the existing result from present literature work.

Pseudo codes of met-heuristic algorithm and machine learning algorithm are shown in appendices II

Flow chart explaining the working of ML algorithm and TLBO algorithm



CHAPTER-4

ANALYTICAL PROGRAM

The three most common benchmark problems are selected. A 10-bar cantilever truss, 25 bar space truss also known as a transmission tower, and a 72 bar space truss consisting of multi-story.

4.1 **A ten bar truss:** the cross-section area of the truss was taken as a continuous variable ranging from 0.1 in^2 to 35.0 in^2. The unit weight of the material was taken as 0.1 lb/in^2 and the modulus of elasticity was taken as 10^7. The allowable maximum stress in compression and tension in any member of the truss was taken to be 25000 psi with a maximum deflection in vertical as well as the horizontal direction was taken as 2.0 inches at any node. The configuration is shown in figure 9. MATLAB script for the geometry and properties of 10-bar truss is developed and is shown in appendices I.

Table 1. Coordinates for 10 bar truss

Joints	*x* (inches)	*y* (inches)	*z* (inches)
1	720	360	0
2	720	0	0
3	360	360	0
4	360	0	0
5	0	360	0
6	0	0	0

Table 2. Load case for 10-bar truss

Node	F_x (psi)	F_y (psi)	F_z (psi)
2	0	-100000	0
4	0	-100000	0

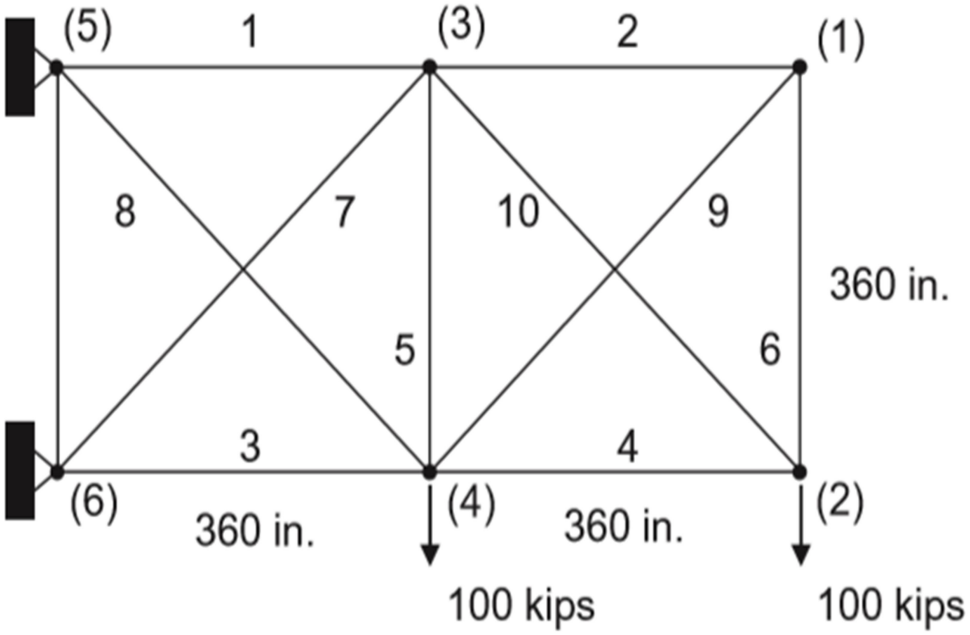


Figure 9. Configuration of 10-bar cantilever truss

4.2 **A twenty-five bar truss:** the cross-section area of the truss was taken as a continuous variable ranging from 0.1 in^2 to 3.4 in^2. The unit weight of the material was taken as 0.1 lb/in^2 and the modulus of elasticity was taken as 10^7. The allowable maximum stress in compression and tension in any member of truss was taken to be 40 ksi with a maximum deflection in x y and z direction was taken as 0.35 inches at any

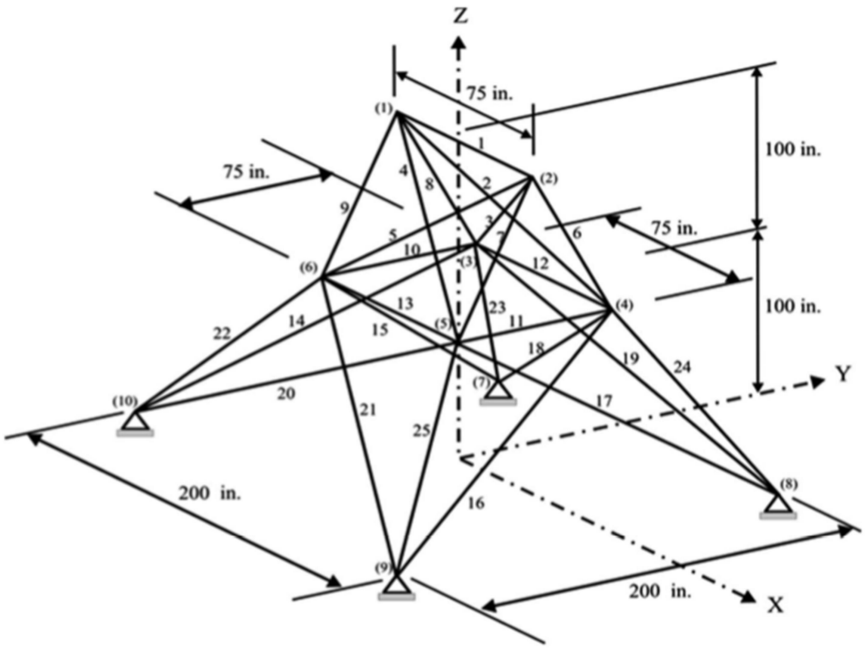


Figure 10. Configuration of 25-bar truss

node. The configuration is shown in the figure 10. MATLAB script for the geometry and properties of 72-bar truss is developed and is shown appendices I.

Table 3. Coordinates for 25-bar truss

Joints	*x* (inches)	*y* (inches)	*z* (inches)
1	-37.5	0.0	200
2	37.5	0.0	200
3	-37.5	37.5	100
4	37.5	37.5	100
5	37.5	-37.5	100
6	-37.5	-37.5	100
7	-100	100	0.0
8	100	100	0.0
9	100	-100	0.0
10	-100	-100	0.0

Table 4. Load Case for 25- bar truss

Node	F_x (psi)	F_y (psi)	F_z (psi)
1	1000	-10000	10000
2	0	-10000	-10000
3	500	0	0
6	600	0	0

4.3 **A seventy-two bar truss:** the cross-section area of the truss was taken as a continuous variable ranging from 0.01 in^2 to 3.0 in^2. The unit weight of the material was taken as 0.1 lb/in^2 and the modulus of elasticity was taken as 10^7. The allowable maximum stress in compression and tension in any member of the truss was taken to be 25 ksi with a maximum deflection in x y and z-direction was taken as 0.25 inches at any node. The configuration is shown in figure 12. MATLAB script for the geometry and properties of 72-bar truss is developed and is shown appendices I.

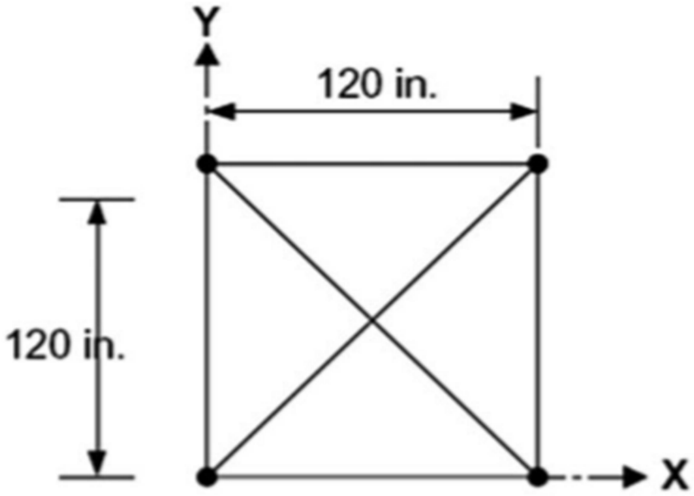


Figure 11. Dimension of 72-bar truss

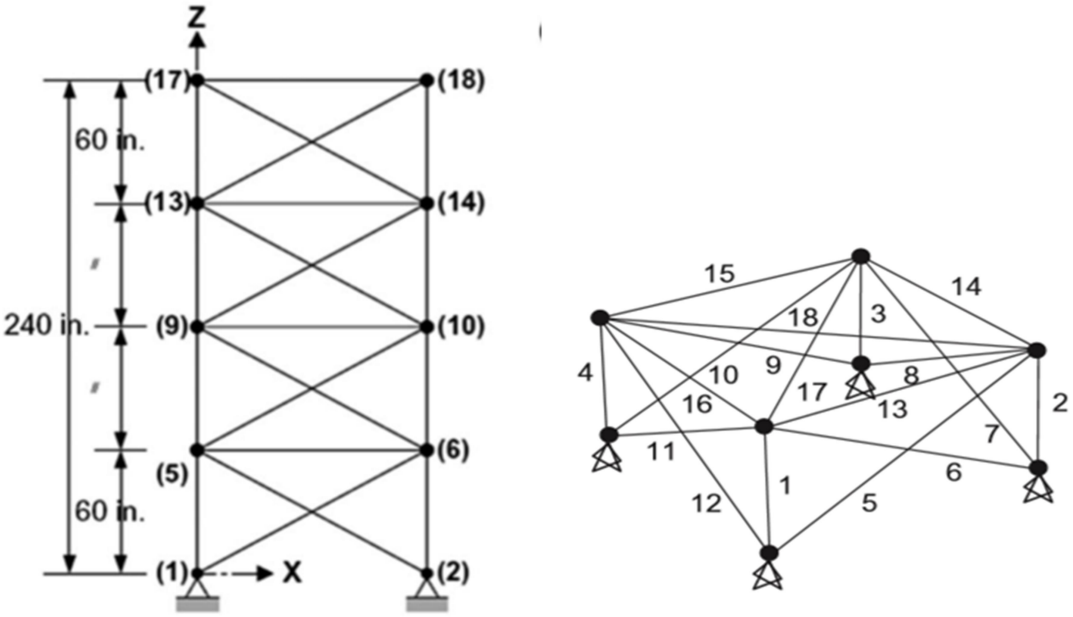


Figure 12. Configuration of 72 bar truss with element and nodal numbering

Table 5. Load case for 72-bar truss

Node	F_x (psi)	F_y (psi)	F_z (psi)
17	5000	5000	-5000

Table 6. Coordinates for 72-bar truss

Joints	*x* (inches)	*y* (inches)	z (inches)
1	0	0	0
2	120	0	0
3	120	120	0
4	0	120	0
5	0	0	60
6	120	0	60
7	120	120	60
8	0	120	60
9	0	0	120
10	120	0	120
11	120	120	120
12	0	120	120
13	0	0	180
14	120	0	180
15	120	120	180
16	0	120	180
17	0	0	240
18	120	0	240
19	120	120	240
20	0	120	240

CHAPTER-5

RESULTS

The result generated by the algorithm in this study was compared with other prominent meta-heuristic approaches in various literature work in terms of total weight, constraints violation, and no. of structural analysis for the benchmark problems. The comparison was done with the modified version of TLBO, Harmony search and its modified version, a modified version of an artificial bee colony, modified version of particle swarm optimization, modified version of big band-big crunch, and modified version of ant colony optimization in the research work of Baghalni et al. (TLBO-MS) [21], Camp and Farshchin (MTLBO-PF) [26], Degertekin and Hayalioglu (TLBO-FBM) [27], Degertekin et al. (SAHS) [28], Kaveh et al.(HBB-BC) [44], Li et al. (HPSO) [8], Camp (BB-BC) [45], Kaveh et al HPSACO [25], Perez et al.(PSO) [9], Lee and Geem (HS) [43].

For the 10 bar, cantilever truss machine learning algorithm generated a viable solution that was slightly heavy than the result of the modified version of TLBO but performed better than harmony search, artificial bee colony, particle search optimization. When compared in terms of structural analysis ML took 4200 analysis which was more than TLBO-MS but it was less than other modified versions of TLBO, HS, ABC, PSO, BB-BC, ACO. In this case, the ant colony gave way less weight but violated the constraint as in table 7.

For 25 bar transmission tower truss the ML algorithm gave a structurally acceptable result which was slightly better than the modified version of TLBO, HS, ABC, PSO, BB-BC, ACO. Machine learning took 1351 structural analysis which was minimum than all other methods as shown in table 8.

For 72 bar multi-story truss ML took 1575 analysis which was less than modified versions of TLBO, HS, ABC, PSO, BB-BC, ACO. The machine learning algorithm gave a structurally acceptable result which was slightly better than modified versions of TLBO, HS, ABC, PSO, BB-BC, ACO as mentioned in table 9.

Table 7. Result of 10-bar truss

10 bar truss result	Li et al. (HPSO) [8]	Kaveh et al. (HPSACO) [25]	Sonmez et al. (ABC-AP) [12]	Degertekin et al.(SAHS) [28]	Degertekin and Hayalioglu (TLBO-FBM) [27]	Camp and Farshchin (MTLBO-PF) [26]	Baghalni et al (TLBO-MS) [21]	This study
Weight (lb)	5060.92	5056.56	5060.88	5061.42	5060.96	5060.973	5060.850	5060.994
W_{avg}(lb)	N/A	N/A	N/A	5061.95	5062.08	5064.808	5060.994	-
Constraint violation (%)	None	0.099	None	None	None	None	None	None
$N_{analysis}$	1,25,000	10,650	500,000	7081	16872	13767	3400	4200

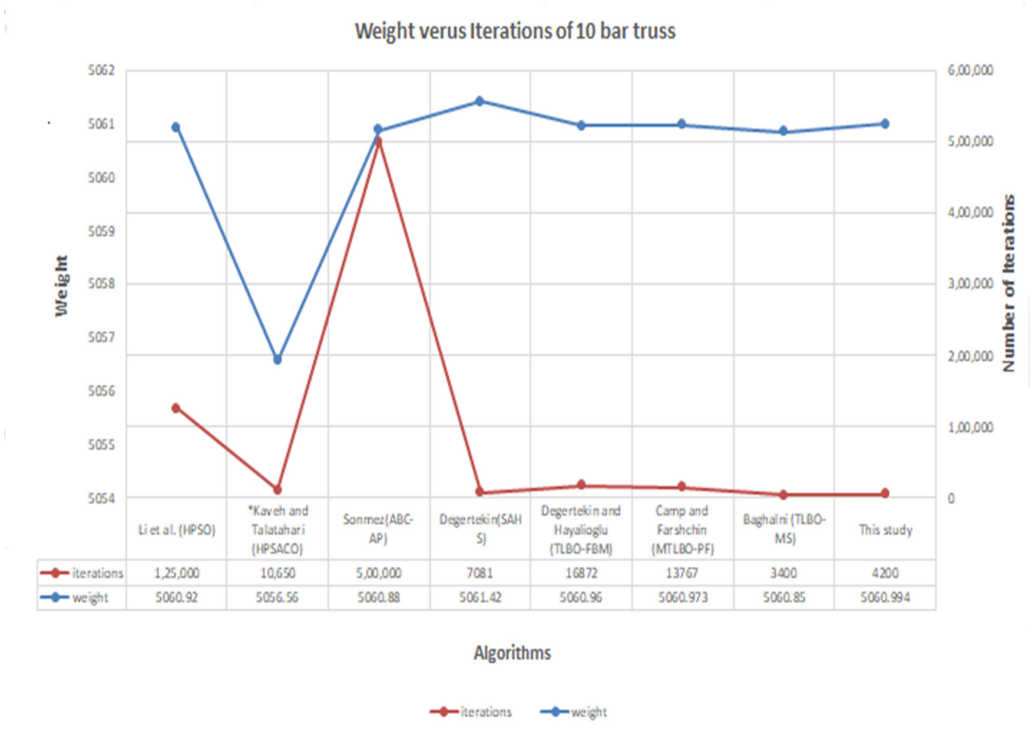


Figure 13. Graphical representation of 10-bar truss

Table 8. Result of 25-bar truss

25 bar truss result	Li et al. (HPSO) [8]	Kaveh et al.		Sonmez et al. (ABC-AP) [12]	Degertekin et al. (SAHS) [28]	Degertekin and Hayaliogl u (TLBO-FBM) [27]	Camp and Farshc hin (MTLBO -PF)[26]	Baghal ni et al. (TLBO -MS) [21]	This study
		HBB -BC [44]	HPSA CO [25]						
Weight (lb)	545.1 9	545. 16	544.99	545.1 9	545.12	545.09	545.17 5	545.16 32	544.74 39
W_{avg}(lb)	N/A	545. 66	545.52	N/A	545.95	545.41	545.48 3	545.18 93	-
Constra int violatio n (%)	None	2.06	3.52	N/A	2.980	2.746	None	None	None
$N_{analysis}$	1,25,0 00	12,5 00	9,875	5,00,0 00	9,051	15,318	12,199	2,200	1351

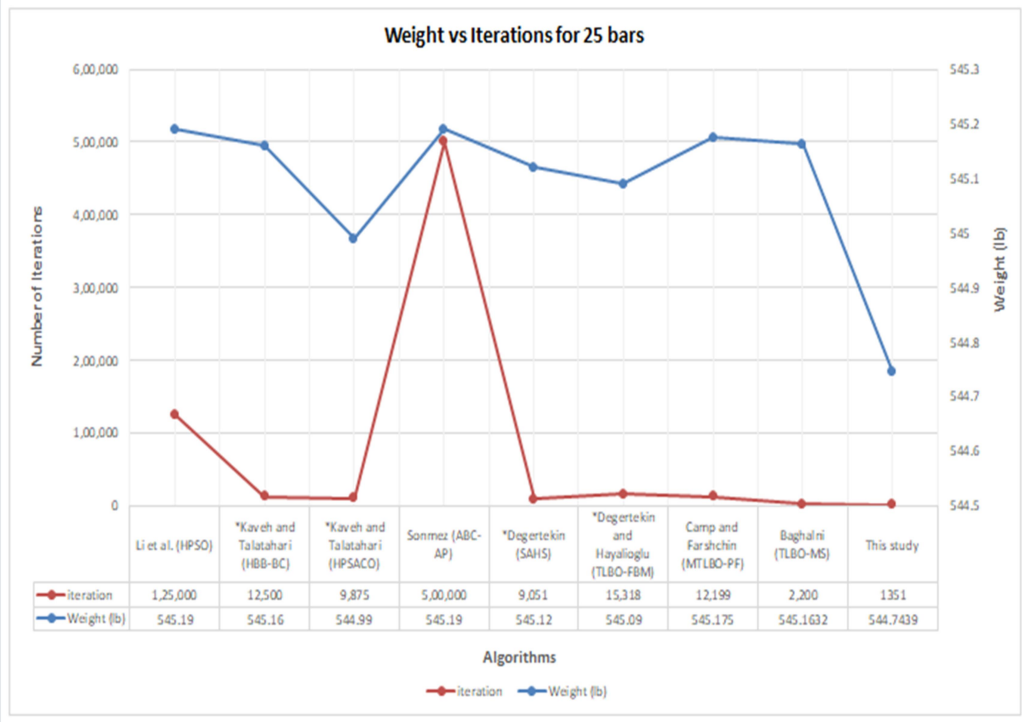


Figure 14. Graphical representation of 25-bar truss

Table 9. Result of 72-bar truss

72 bar truss result	Lee and Geem (HS) [43]	Perez et al.(PSO) [9]	Camp (BB-BC) [45]	Li et al. (HPSO) [8]	Kaveh et al.(HBB-BC) [44]	Degertekin et al.(SAHS) [28]	Degertekin and Hayalioglu (TLBO-FBM) [27]	Camp and Farshchin (MTLBO-PF)[26]	Baghalni et al.(TLBO-MS) [21]	This study
Weight (lb)	379.27	381.91	379.85	369.65	379.66	380.62	379.63	379.632	379.617	379.5261
W_{avg}(lb)	N/A	N/A	382.08	N/A	N/A	382.62	380.2	379.759	379.769	-
Constraint violation (%)	0.217	None	None	39.075	None	0.259	None	None	None	None
$N_{analysis}$	20,000	8,000	19,621	12,5000	13,200	13,742	19,709	21,542	10,000	1575

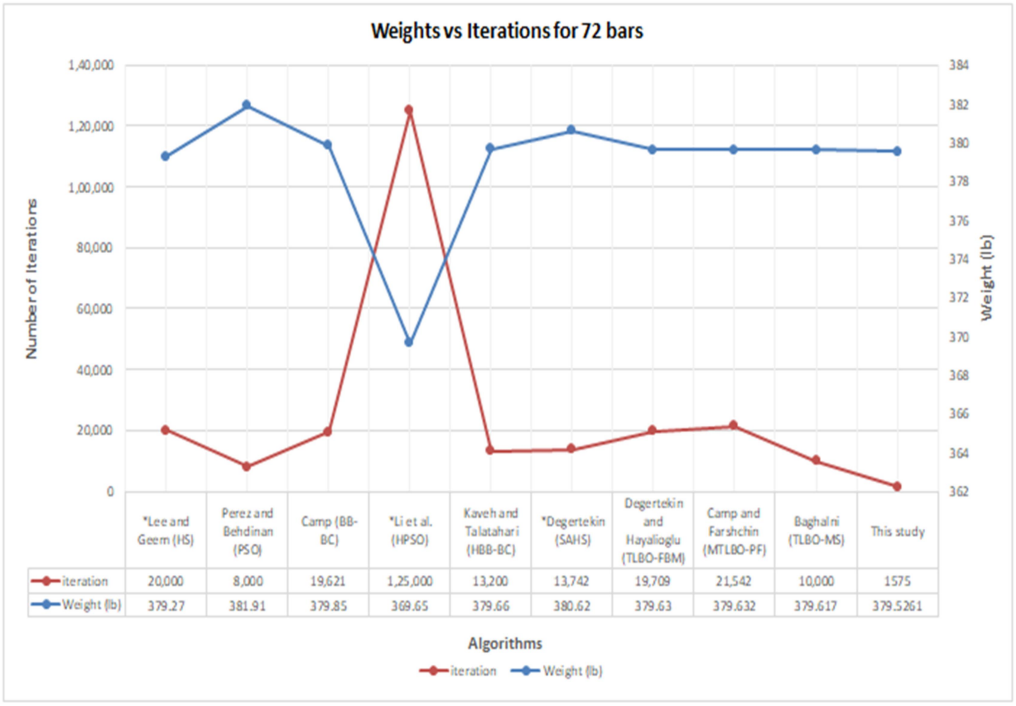


Figure 15. Graphical representation for 72-bar truss

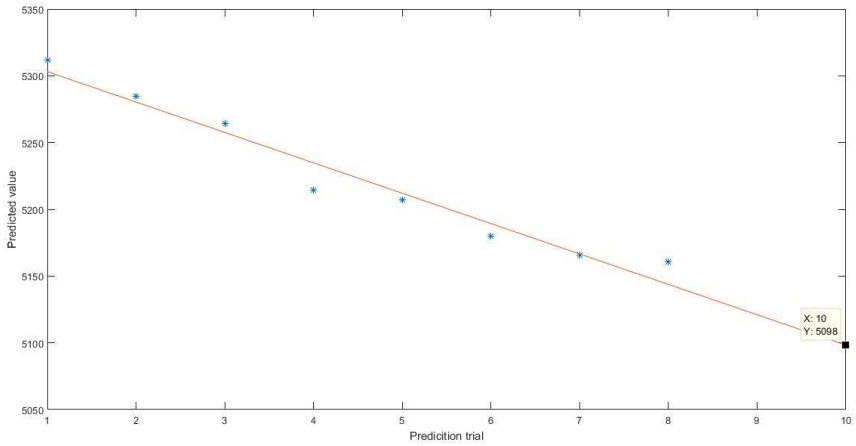


Figure 16. Convergence history of ML algorithm for 10-bar truss

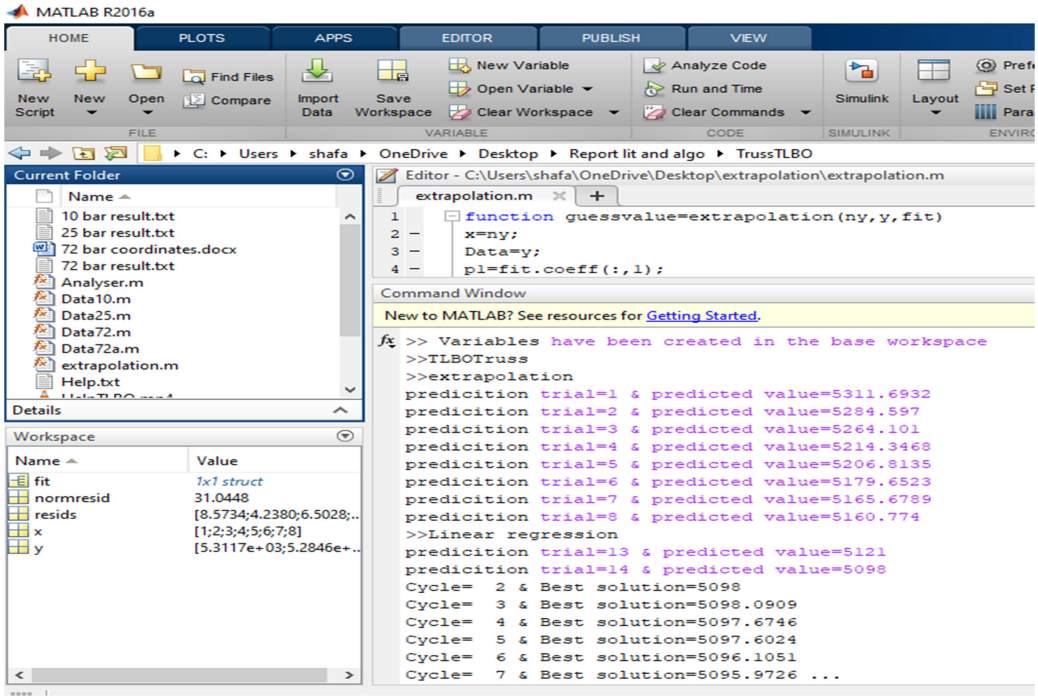


Figure 17. Output generated by ML and TLBO algorithm for 10-bar truss

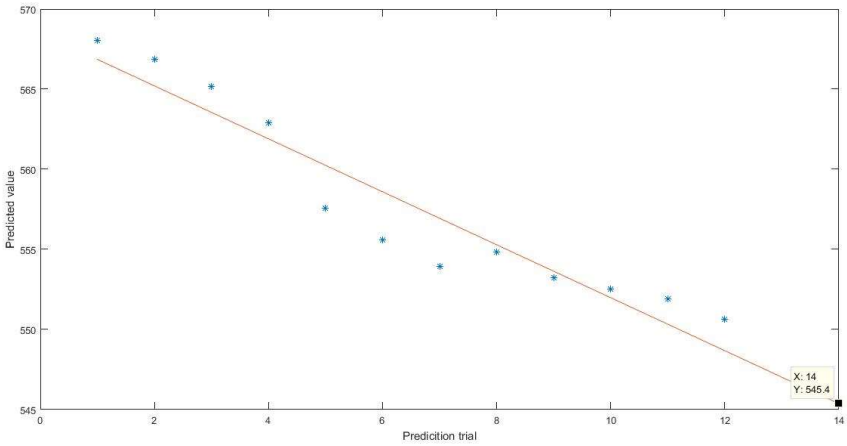


Figure 18. Convergence history of ML algorithm for 25-bar truss

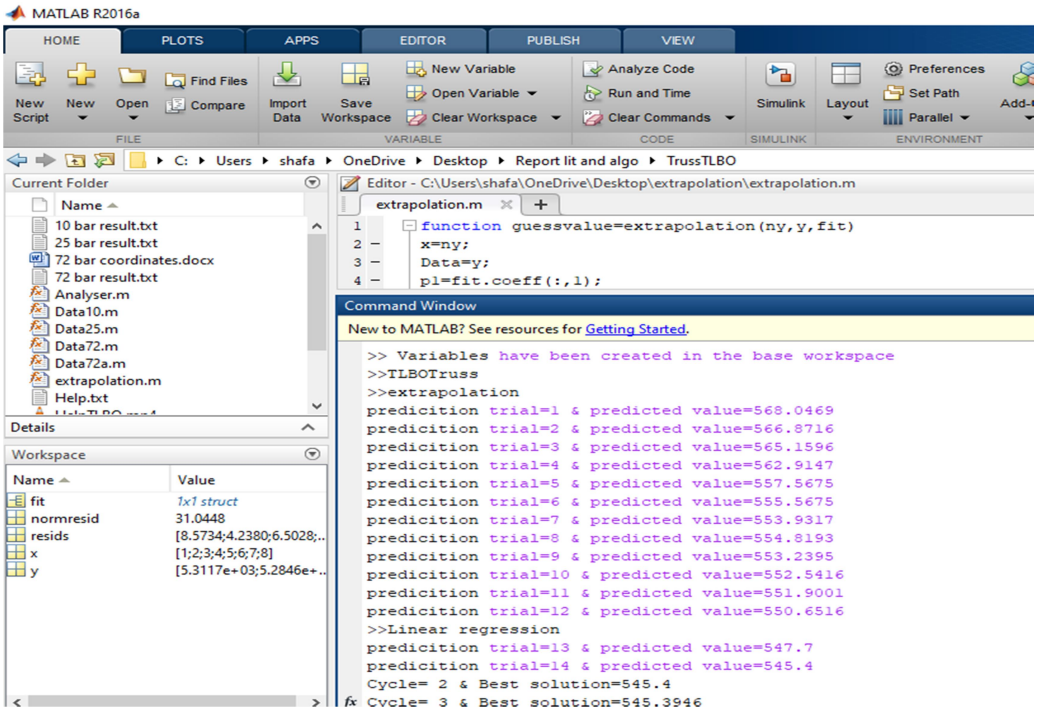


Figure 19.Output generated by ML and TLBO algorithm for 25-bar truss

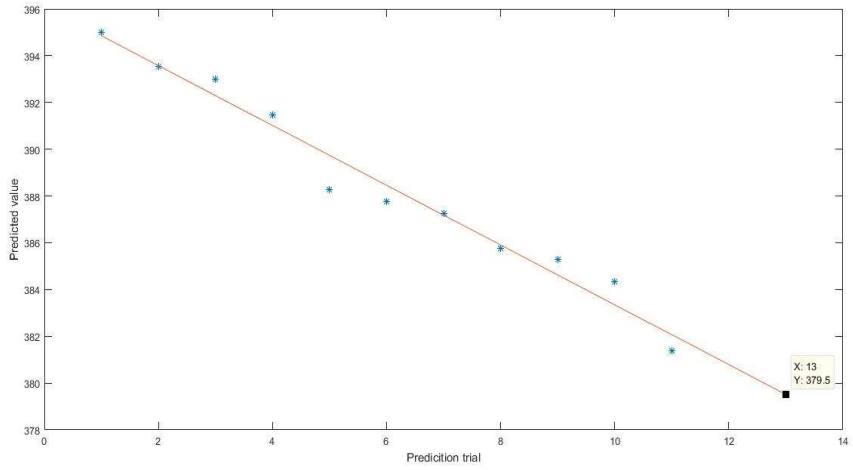


Figure 20.Convergence history of ML algorithm for 72-bar truss

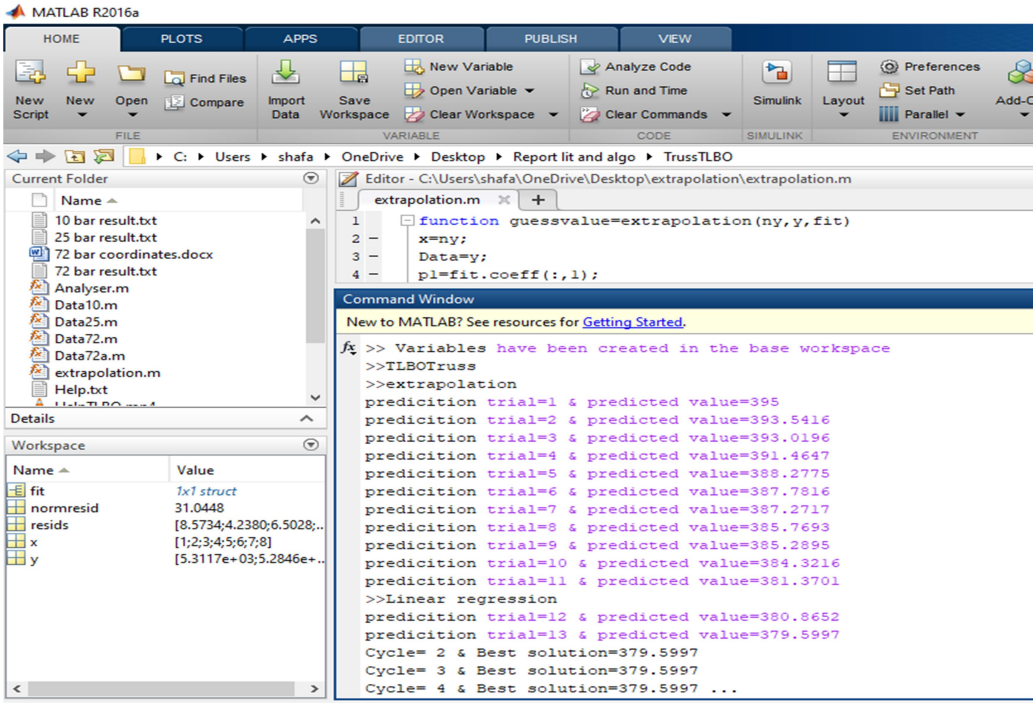


Figure 21. Output generated by ML and TLBO algorithm for 72-bar truss

CHAPTER-6

CONCLUSION

Conclusion and future scope

The machine-learning algorithm showed no constraint violation whereas modified versions of TLBO, HS, ABC, PSO, BB-BC, ACO violated the constraint at some point of analysis when pushed to the optimization limit. In this research work, three benchmark problems were optimized and the machine learning algorithm successfully optimized them and generated a feasible design which was as good as the compared methods. The advantage of ML comes in as the intelligent and adaptive way to reduce the search space which in turn reduces the structural analysis needed to reach the desired solution. ML algorithm outperforms the other meta-heuristic algorithms and other variants in terms of robustness and always leads to a fully feasible solution. In future research, machine learning can be applied to other structural optimization and it can be used with other incapable optimization algorithms such as gradient-based methods and some meta-heuristics methods to eliminate the issues faced by them. In the future other complex and adaptive machine learning methods can use in the field of structural engineering as these research works prove the fact the machine learning can be used in the field to the structural design field.

REFERENCES

[1] Christensen, P. W., & Klarbring, A. (2008). *An introduction to structural optimization* (Vol. 153). Springer Science & Business Media.

[2] Salehi, H., & Burgueno, R. (2018). Emerging artificial intelligence methods in structural engineering. Engineering structures, 171, 170-189.

[3] http://www.mathworks.com/matlabcentral/fileexchange

[4] Zhengtong, H., Zhengqi, G., Xiaokui, M., & Wanglin, C. (2019). Multimaterial layout optimization of truss structures via an improved particle swarm optimization algorithm. Computers & Structures, 222, 10-24.

[5] Mortazavi, A., Toğan, V., & Moloodpoor, M. (2019). Solution of structural and mathematical optimization problems using a new hybrid swarm intelligence optimization algorithm. Advances in Engineering Software, 127, 106-123.

[6] Kaveh, A., Sheikholeslami, R., Talatahari, S., & Keshvari-Ilkhichi, M. (2014). Chaotic swarming of particles: A new method for size optimization of truss structures. Advances in Engineering Software, 67, 136-147.

[7] Gomes, H. M. (2011). Truss optimization with dynamic constraints using a particle Li, L. J., Huang, Z. B., Liu, F., & Wu, Q. H. (2007). A heuristic particle swarm optimizer for optimization of pin connected structures. Computers & Structures, 85(7-8), 340-349.

[8] Li, L. J., Huang, Z. B., Liu, F., & Wu, Q. H. (2007). A heuristic particle swarm optimizer for optimization of pin connected structures. Computers & Structures, 85(7-8), 340-349.

[9] Perez, R. L., & Behdinan, K. (2007). Particle swarm approach for structural design optimization. Computers & Structures, 85(19-20), 1579-1588.

[10] Camp, C. V., Bichon, B. J., & Stovall, S. P. (2005). Design of steel frames using ant colony optimization. Journal of Structural Engineering, 131(3), 369-379.

[11] Camp, C. V., & Bichon, B. J. (2004). Design of space trusses using ant colony optimization. Journal of structural engineering, 130(5), 741-751.

[12] Sonmez, M. (2011). Artificial Bee Colony algorithm for optimization of truss structures. Applied Soft Computing, 11(2), 2406-2418.

[13] Kaveh, A., & Ghazaan, M. I. (2017). Enhanced whale optimization algorithm for sizing optimization of skeletal structures. Mechanics Based Design of Structures and Machines, 45(3), 345-362.

[14] Bureerat, S., & Pholdee, N. (2016). Optimal truss sizing using an adaptive differential evolution algorithm. Journal of Computing in Civil Engineering, 30(2), 04015019.

[15] Ahrari, A., & Atai, A. A. (2013). Fully stressed design evolution strategy for shape and size optimization of truss structures. Computers & Structures, 123, 58-67.

[16] Zawidzki, M., & Jankowski, Ł. (2019). Multiobjective optimization of modular structures: Weight versus geometric versatility in a Truss-Z system. Computer-Aided Civil and Infrastructure Engineering, 34(11), 1026-1040.

[17] Assimi, H., Jamali, A., & Nariman-Zadeh, N. (2017). Sizing and topology optimization of truss structures using genetic programming. Swarm and evolutionary computation, 37, 90-103.

[18] Toğan, V., & Daloğlu, A. T. (2006). Optimization of 3d trusses with adaptive approach in genetic algorithms. Engineering Structures, 28(7), 1019-1027.

[19] Cazacu, R., & Grama, L. (2014). Steel truss optimization using genetic algorithms and FEA. Procedia Technology, 12, 339-346.

[20] Jenkins, W. M. (1991). Towards structural optimization via the genetic algorithm. Computers & Structures, 40(5), 1321-1327.

[21] Baghlani, A., Makiabadi, M. H., & Maheri, M. R. (2017). Sizing optimization of truss structures by an efficient constraint-handling strategy in TLBO. Journal of Computing in Civil Engineering, 31(4), 04017004.

[22] Tejani, G. G., Savsani, V. J., Patel, V. K., & Savsani, P. V. (2018). Size, shape, and topology optimization of planar and space trusses using mutation-based

improved metaheuristics. Journal of Computational Design and Engineering, 5(2), 198-214.

[23] Savsani, V. J., Tejani, G. G., Patel, V. K., & Savsani, P. (2017). Modified meta-heuristics using random mutation for truss topology optimization with static and dynamic constraints. Journal of Computational Design and Engineering, 4(2), 106-130

[24] Farshchin, M., Camp, C. V., & Maniat, M. (2016). Multi-class teaching–learning-based optimization for truss design with frequency constraints. Engineering Structures, 106, 355-369.

[25] Kaveh, A., & Talatahari, S. (2009). Particle swarm optimizer, ant colony strategy and harmony search scheme hybridized for optimization of truss structures. Computers & Structures, 87(5-6), 267-283.

[26] Camp, C. V., & Farshchin, M. (2014). Design of space trusses using modified teaching–learning based optimization. Engineering Structures, 62, 87-97.

[27] Degertekin, S. O., & Hayalioglu, M. S. (2013). Sizing truss structures using teaching-learning-based optimization. Computers & Structures, 119, 177-188.

[28] Degertekin, S. O. (2012). Improved harmony search algorithms for sizing optimization of truss structures. Computers & Structures, 92, 229-241.

[29] Toğan, V. (2012). Design of planar steel frames using teaching–learning based optimization. Engineering Structures, 34, 225-232.

[30] Tejani, G. G., Pholdee, N., Bureerat, S., Prayogo, D., & Gandomi, A. H. (2019). Structural optimization using multi-objective modified adaptive symbiotic organisms search. Expert Systems with Applications, 125, 425-441.

[31] Tejani, G. G., Savsani, V. J., Bureerat, S., & Patel, V. K. (2017). Topology and size optimization of trusses with static and dynamic bounds by modified symbiotic organisms search. Journal of Computing in Civil Engineering, 32(2), 04017085.

[32] Afshari, H., Hare, W., & Tesfamariam, S. (2019). Constrained multi-objective optimization algorithms: Review and comparison with application in reinforced concrete structures. Applied Soft Computing, 83, 105631.

[33] Dede, T., Grzywiński, M., & Rao, R. V. (2020). Jaya: A New Meta-heuristic Algorithm for the Optimization of Braced Dome Structures. In Advanced Engineering Optimization Through Intelligent Techniques (pp. 13-20). Springer, Singapore.

[34] Degertekin, S. O., Lamberti, L., & Ugur, I. B. (2019). Discrete sizing/layout/topology optimization of truss structures with an advanced Jaya algorithm. Applied Soft Computing, 79, 363-390.

[35] Degertekin, S. O., Lamberti, L., & Ugur, I. B. (2018). Sizing, layout and topology design optimization of truss structures using the Jaya algorithm. Applied Soft Computing, 70, 903-928.

[36] Nwe, May Thu Nwe, and James K. Guest. "Topology Optimization of Truss Structures Considering Stress and Stability Constraints." Structures Congress 2019: Buildings and Natural Disasters. Reston, VA: American Society of Civil Engineers, 2019.

[37] Aksöz, Z., & Preisinger, C. (2019, September). An Interactive Structural Optimization of Space Frame Structures Using Machine Learning. In Design Modelling Symposium Berlin (pp. 18-31). Springer, Cham.

[38] Greco, L. (2018, July). Machine Learning and Optimization techniques for Steel Connections. In Proceedings of IASS Annual Symposia (Vol. 2018, No. 23, pp. 1-8). International Association for Shell and Spatial Structures (IASS).

[39] Bekas, G., & Stavroulakis, G. (2017). Machine Learning and Optimality in Multi Storey Reinforced Concrete Frames. Infrastructures, 2(2), 6.

[40] Tamura, T., Ohsaki, M., & Takagi, J. (2018). Machine learning for combinatorial optimization of brace placement of steel frames. Japan Architectural Review, 1(4), 419-430.

[41] Parvin, A., & Serpen, G. (1999). Recurrent neural networks for structural optimization. Computer-Aided Civil and Infrastructure Engineering, 14(6), 445-451.

[42] Papadrakakis, M., Lagaros, N. D., & Tsompanakis, Y. (1998). Structural optimization using evolution strategies and neural networks. Computer methods in applied mechanics and engineering, 156(1-4), 309-333.

[43] Lee, K. S., & Geem, Z. W. (2004). A new structural optimization method based on the harmony search algorithm. Computers & structures, 82(9-10), 781-798.

[44] Kaveh, A., & Talatahari, S. (2009). Size optimization of space trusses using Big Bang–Big Crunch algorithm. Computers & structures, 87(17-18), 1129-1140.

[45] Camp, C. V. (2007). Design of space trusses using Big Bang–Big Crunch optimization. Journal of Structural Engineering, 133(7), 999-1008.

[46] Rao, R. V., Savsani, V. J., & Vakharia, D. P. (2011). Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. Computer-Aided Design, 43(3), 303-315.

[47] Siddique, N., & Adeli, H. (2013). Computational intelligence: synergies of fuzzy logic, neural networks and evolutionary computing. John Wiley & Sons.

APPENDICES-I

The MATLAB script for the geometry and properties of benchmark truss problem is shown below.

For 10-bar cantilever truss:

```
function D=Data10
% Design parameters for the benchmark 10-bar truss problem
Coord=360*[2 1 0;2 0 0;1 1 0;1 0 0;0 1 0;0 0 0];
Con=[5 3;1 3;6 4;4 2;3 4;1 2;6 3;5 4;4 1;3 2];
Re=[0 0 1;0 0 1;0 0 1;0 0 1;1 1 1;1 1 1];
Load=zeros(size(Coord));Load(2,:)=[0  -1e5  0];Load(4,:)=[0
-1e5 0];
E=ones(1,size(Con,1))*1e7;
A=ones(1,10);
% Available sections
AV=[0.1:0.1:35];%in^2
%Allowable Stress
TM=25000;%psi
%Allowable Displacement
DM=2;%inch
%Density
RO=.1;%lb/in^3
LB=ones(1,10)*0.1;
UB=ones(1,10)*33;
D=struct('Coord',Coord,'Con',Con,'Re',Re,'Load',Load,'E',E
','A',A,'AV',AV,'TM',TM,'DM',DM,'RO',RO,'LB',LB,'UB',UB
);
```

For 25 bar transmission tower truss:

```
function D=Data25
Coord=[-37.5 0 200;37.5 0 200;-37.5 37.5 100;37.5 37.5 100;37.5 -37.5 100;-37.5 -37.5 100;-100 100 0;100 100 0;100 -100 0;-100 -100 0];
Con=[1 2;1 4;2 3;1 5;2 6;2 4;2 5;1 3;1 6;3 6;4 5;3 4;5 6;3 10;6 7;4 9;5 8;4 7;3 8;5 10;6 9;6 10;3 7;4 8;5 9];
Re=[0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;1 1 1;1 1 1;1 1 1;1 1 1];
Load=zeros(size(Coord));Load(1,:)=[1e3 -1e4 -1e4];Load(2,:)=[0 -1e4 -1e4];Load(3,:)=[5e2 0 0];Load(6,:)=[6e2 0 0];
E=ones(1,size(Con,1))*1e7;
A=ones(1,25);
% Available sections
AV=[0.1:0.1:3.4];%in^2
%Allowable Stress
TM=40000;%psi
%Allowable Displacement
DM=0.35;%inch
%Density
RO=.1;%lb/in^3
LB=ones(1,25)*0.1;
UB=ones(1,25)*3.4;
D=struct('Coord',Coord,'Con',Con,'Re',Re,'Load',Load,'E','E','A','A','AV',AV,'TM',TM,'DM',DM,'RO',RO,'LB',LB,'UB',UB);
```

For 72-bar multi storey truss structure:

```
function D=Data72
Coord=[0 0 0;120 0 0;120 120 0;0 120 0;0 0 60;120 0 60;120 120 60;0 120 60;0 0 120;120 0 120;120 120 120;0 120 120;0 0 180;120 0 180;120 120 180; 0 120 180;0 0 240;120 0 240;120 120 240;0 120 420];
Con=[1 5;1 6;1 8;2 6;2 5;2 7;3 7;3 8;3 6;4 8;4 7;4 5;5 6;6 7;7 8;8 5;5 7; 6 8;5 9;5 10;5 12;6 10;6 9;6 11;7 11;7 12;7 10;8 12;8 11;8 9;9 10;10 11; 11 12;12 9;9 11;10 12;9 13,9 14;9 16;10 14;10 13;10 15;11 15;11 16;11 14; 12 16;12 15;12 13;13 14;14 15;15 16;16 13;13 15;14 16;13 17;13 18;13 20;14 18;14 17;14 19;15 19;15 20;15 18;16 20;16 19;16 17;17 18;18 19;19 20;20 17;17 19;18 20];
Re=[1 1 1;1 1 1;1 1 1;1 1 1;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0;0 0 0];
Load=zeros(size(Coord));Load(17,:)=[5e3 5e3 -5e3];
E=ones(1,size(Con,1))*1e7;
A=ones(1,72);
% Available sections
AV=[0.1:0.1:3];%in^2
%Allowable Stress
TM=25000;%psi
%Allowable Displacement
DM=0.25;%inch
%Density
RO=.1;%lb/in^3
LB=ones(1,72)*0.1;UB=ones(1,72)*3;
D=struct('Coord',Coord','Con',Con','Re',Re','Load',Load','E',E','A',A','AV',AV','TM',TM','DM',DM','RO',RO','LB',LB,'UB',UB);
```

APPENDICES-II

The pseudo code for TLBO algorithm along with analyser function algorithm is shown below.

```
function TLBOTruss
global D

%% Optimization parameters
D=Data10;                                   % Read design parameters of the
benchmark 10-bar truss from Data file
TL.Itmax=200;                               % Maximum number of iterations
TL.PopSize=75;                              % Population size
TL.LB=D.LB;                                 % Lowerbound row vector (LB)
TL.UB=D.UB;                                 % Upperbound row vector (UB)
%% Applies Machine Learning to give design
function guessvalue=extrapolation(ny,y,fit)
%% Randomely generate initial designs between LB and UB
function INITIAL
Cycle=1;
for I=1:TL.PopSize
   Designs(I,:)=TL.LB+rand(1,size(TL.LB,2)).*(TL.UB-TL.LB);          % Row vector
end

%% TLBO main loop
Best{1}=[];
for Cycle=2:TL.Itmax
   %% Evaluate the designs generated in the previous iteration
   [PObj,Obj]=Analyser(Designs);
   %% Specify best designs and keep them
   [Best{Cycle},Designs,PObj,WMeanPos]=Specifier(PObj,Obj,Designs,Best{Cycle-
1});
   %% Apply Teaching

[Designs,PObj,Obj]=Teaching(TL,Designs,PObj,Obj,Best{Cycle}.GBest.Design,WMea
nPos);
   [Best{Cycle},Designs,PObj,WMeanPos]=Specifier(PObj,Obj,Designs,Best{Cycle-
1});
   %% Apply Learning
   [Designs]=Learning(TL,Designs,PObj);
   % Plot time history of the best solution vs. iteration
   hold on;plot(Cycle,Best{Cycle}.GBest.PObj,'b*');xlabel('Iteration');ylabel('Best
solution');pause(0.0001)
   % Print current best solution
   fprintf('Cycle=%3d & Best solution=%6.8g\n',Cycle,Best{Cycle}.GBest.PObj)
end

%% Save results
```

```
save('TimeHist.mat','Best')
%.....%

function [PObj,Obj]=Analyser(Designs)

global D
% This function evaluates the generated designs
for I=1:size(Designs,1)
   D.A=Designs(I,:);
   [~,~,~,Obj(I,1),PObj(I,1)]=ST(D); %Obj: Objective value;   PObj: Penalized
objecive value
end

end
function [F,U,R,WE,GOAL]=ST(D)

D.A=D.A';
w=size(D.Re);S=zeros(3*w(2));U=1-D.Re;f=find(U);
WE=0;
for i=1:size(D.Con,2)
   H=D.Con(:,i);C=D.Coord(:,H(2))-D.Coord(:,H(1));Le=norm(C);
   T=C/Le;s=T*T';G=D.E(i)*D.A(i)/Le;Tj(:,i)=G*T;
   e=[3*H(1)-2:3*H(1),3*H(2)-2:3*H(2)];S(e,e)=S(e,e)+G*[s -s;-s s];
   WE=WE+Le*D.A(i)*D.RO;
end
U(f)=S(f,f)\D.Load(f);F=sum(Tj.*(U(:,D.Con(2,:))-U(:,D.Con(1,:))));
R=reshape(S*U(:),w);R(f)=0;
TS=(((abs(F'))./D.A)/D.TM)-1;%Tension
US=abs(U')/D.DM-1;%Displacement
PS=sum(TS.*(TS>0));
PD=sum(sum(US.*(US>0)));
GOAL=WE*(1+PS+PD)^2;% Penalty function
end
%.....%

function [Designs,PObj,Obj]=Teaching(TL,Designs,PObj,Obj,Teacher,WMeanPos)



for I=1:size(Designs,1)
   TF=randi([1,2],1,size(Teacher,2));
   Diff_Mean=rand(1,size(Teacher,2)).*TF.*(Teacher-WMeanPos);
   NewDesigns(I,:)=Designs(I,:)+sign(Teacher-Designs(I,:)).*abs(Diff_Mean);
end

for i=1:size(Designs,1)
```

```
  for j=1:size(Teacher,2)
    if NewDesigns(i,j)<TL.LB(1,j)
      NewDesigns(i,j)=TL.LB(1,j);
    elseif NewDesigns(i,j)>TL.UB(1,j)
      NewDesigns(i,j)=TL.UB(1,j);
    end
  end
end

%% Evaluate the new designs
[NPObj,NObj]=Analyser(NewDesigns);

%% Check individuals for improvement and update the designs that are improved
for I=1:size(Designs,1)
  if NPObj(I,1)<PObj(I,1)
    PObj(I,1)=NPObj(I,1);
    Obj(I,1)=NObj(I,1);
    Designs(I,:)=NewDesigns(I,:);
  end
end


%.....%

function [Designs,PObj,Obj]=Teaching(TL,Designs,PObj,Obj,Teacher,WMeanPos)
%% This function applies teaching phase


for I=1:size(Designs,1)
  TF=randi([1,2],1,size(Teacher,2));
  Diff_Mean=rand(1,size(Teacher,2)).*TF.*(Teacher-WMeanPos);
  NewDesigns(I,:)=Designs(I,:)+sign(Teacher-Designs(I,:)).*abs(Diff_Mean);
end

%% Check feasibility of the designs
for i=1:size(Designs,1)
  for j=1:size(Teacher,2)
    if NewDesigns(i,j)<TL.LB(1,j)
      NewDesigns(i,j)=TL.LB(1,j);
    elseif NewDesigns(i,j)>TL.UB(1,j)
      NewDesigns(i,j)=TL.UB(1,j);
    end
  end
end

%% Evaluate the new designs
```

```
[NPObj,NObj]=Analyser(NewDesigns);

%% Check individuals for improvement and update the designs that are improved
for I=1:size(Designs,1)
   if NPObj(I,1)<PObj(I,1)
      PObj(I,1)=NPObj(I,1);
      Obj(I,1)=NObj(I,1);
      Designs(I,:)=NewDesigns(I,:);
   end
end

%.....%

function [Designs]=Learning(TL,Designs,PObj)


for I=1:size(Designs,1)
   % Selection
   A=(randperm(size(Designs,1)))';
   First=A(1);
   Second=A(2);
   if PObj(First)>PObj(Second)
      First=A(2);
      Second=A(1);
   end
   BetterDesign=Designs(First,:);
   WorseDesign =Designs(Second,:);
   NewDesigns(I,:)=WorseDesign+rand(1,size(Designs,2)).*(BetterDesign-
WorseDesign);
end
%% Check feasibility of the designs
for i=1:size(Designs,1)
   for j=1:size(Designs,2)
      if NewDesigns(i,j)<TL.LB(1,j)
         NewDesigns(i,j)=TL.LB(1,j);
      elseif NewDesigns(i,j)>TL.UB(1,j)
         NewDesigns(i,j)=TL.UB(1,j);
      end
   end
end

%% Evaluate the new designs
[NPObj,NObj]=Analyser(NewDesigns);

%% Check individuals for improvement and update the designs that are improved
for I=1:size(Designs,1)
```

```
   if NPObj(I,1)<PObj(I,1)
      PObj(I,1)=NPObj(I,1);
      Obj(I,1)=NObj(I,1);
      Designs(I,:)=NewDesigns(I,:);
   end
end
%.....%
function
[NewBest,Designs,PObj,WMeanPos]=Specifier(PObj,Obj,Designs,PreviousBest)


%% Sort new results
[~,b]=sort(PObj);                          % Sort Penalized objective values
Designs=Designs(b,:);                      % Sorted designs
Obj=Obj(b,:);                              % Sorted objectives
PObj=PObj(b,:);                            % Sorted penalized objectives

WMeanPos=mean(Designs);

%% Find the bests
GBest.PObj=PObj(1);                        % Current best
GBest.Obj=Obj(1);                          % Current best
GBest.Design=Designs(1,:);                 % Current best

% Find current best non penalized individual
C=find(PObj==Obj);
if isempty(C)==1
   NP.Obj=[];
   NP.Design=[];
else
   C=C(1);
   NP.Obj=PObj(C);
   NP.Design=Designs(C,:);
end

%% Compare the current bests with previous ones

if isempty(PreviousBest)==1
   NewBest.GBest=GBest;
elseif GBest.PObj<PreviousBest.GBest.PObj
   NewBest.GBest=GBest;
else
   NewBest.GBest=PreviousBest.GBest;
end

if isempty(PreviousBest)==1
```

```
   NewBest.NP=NP;
elseif isempty(PreviousBest.NP.Obj)~=0
   if NP.Obj<PreviousBest.NP.Obj
      NewBest.NP=NP;
   end
else
      NewBest.NP=PreviousBest.NP;
end
%% Keep the best solutions in the population but do not repaeat them more than
once
K=0;
for I=1:size(Designs,1)
   if sum(Designs(I,:)==NewBest.GBest.Design)==size(Designs,2)
      K=K+1;
   end
end

if K==0
   Designs(end,:)=NewBest.GBest.Design;
   PObj(end)=NewBest.GBest.PObj;
end

L=0;
for I=1:size(Designs,1)
   if sum(Designs(I,:)==NewBest.NP.Design)==size(Designs,2)
      L=L+1;
   end
end

if L==0
   Designs(end-1,:)=NewBest.NP.Design;
   PObj(end-1,:)=NewBest.NP.Obj;
end
```

Appendices III

The pseudo code for Machine learning algorithm is shown below.

```
function [PObj,Obj]=function guessvalue
function guessvalue=extrapolation(ny,y,fit)
ntm==size((Con),2);
tdata=xlsread('trussdata.xls','ntm')
selectedsheet=numcmp(nt);
if ntm==nt
x=ny;
Data=y;
p1=fit.coeff(:,1);
p2=fit.coeff(:,2);
X_train=train_set(:,1:end-1); Y_train=train_set(:,end);
X_test=test_set(:,1:end-1); Y_test=test_set(:,end);

Test_mdl = fitlm(X_train,Y_train);
W=Test_mdl.Coefficients{:,1}

predicted_values=predict(Test_mdl,X_test);
mse2=sqrt(mean((predicted_values-Y_test).^2))



X_train=[ones(N,1), X_train];

W=zeros(size(X_train,2),1);
W_old=ones(size(X_train,2),1);

while(norm(W_old-W) > 10^-5)
   W_old=W;
   W = W - 0.1/N*X_train'*(X_train*W - Y_train);
end
predicted_values=[ones(length(X_test),1),X_test]*W;
```

```
mse3=sqrt(mean((predicted_values-Y_test).^2))
 for i=1:x
x=i;
guessvalue(1,i)= p1*x + p2 ;
 end
 figure;
 plot(Data,'*');
 hold on;
 plot(guessvalue,'-')
 fprintf('predicition trial=%3d & predicted value=%6.8g\n',
figure
hold on
scatter(X_test,Y_test)
fplot(W(1)+W(2)*x)
xlabel({'X_1'});
ylabel({'Y'});
title({'Linear Regression '});
xlim([-3 3])
hold off
function [PObj,Obj]=Analyser(Designs)
guessvalue==Best{1};
function
[NewBest,Designs,PObj,WMeanPos]=Specifier(PObj,Obj,Designs,PreviousBest)
end
else
 function [PObj,Obj]=INITIAL
```